Ultralydkommunikasjon i sensornettverk

Martin Dalbro
Masteroppgave i
Elektronikk og Datateknologi
Mikroelektronikk
Universitetet i Oslo

2008

Kapittel 1

Forord

Denne oppgaven avslutter min mastergrad i Elektronikk og Datateknologi, Mikroelektronikk ved Fysisk Institutt, Universitetet i Oslo. Studiet har vært interessant og morsomt, samtidig som det til tider har vært slitsomt og meget arbeidskrevende. Grunnen til at denne oppgaven fristet var at den er praktisk rettet, med mulighet til å teste ut løsningner og ideer i praksis, i motsetning til en mer teoretisk rettet oppgave.

Oppgaven er en del av et større forskningsprosjekt som er støttet av Hydro med tilsammen 10 studenter fra Elektronikk og Datateknologistudiet og Informatikkstudiet ved Institutt for Informatikk. Hydro har stilt med økonomisk støtte til utstyr og utlån av bærbar PC under hele prosjekttiden. Instituttet har stilt med eget arbeidsrom for studentene med oppgave for Hydro. Jeg håper at andre kan dra nytte av resultatene i denne oppgaven og eventuelt bruke den som grunnlag for videre forskning.

Jeg vil takke for god oppfølging av veilederne mine, både hovedveileder Tor-Sverre Lande og medveiledere Oddvar Søråsen, Stein Gjessing og Håvard Kolle Riis. Oppfølgingen med veiledningsmøter underveis gjennom hele prosessen, og muligheten til å titte innom kontoret for en kjapp diskusjon har vært til stor hjelp. Positive tilbakemeldinger, hint i riktig retning og oppmuntringer har gjort det lettere å arbeide videre de gangene arbeidet har vært tungt. Professor Sverre Holm, tilknyttet DSB-gruppa ved Institutt for Informatikk, fortjener også en stor takk. Hans kunnskap om ultralyd og hjelp med å skaffe ultralydtransdusere har vært til stor hjelp i prosjektet. Jeg vil også takke medstudent Aart Joakim in't Veld, som har arbeidet med en tilsvarende oppgave med bruk av infrarødt lys som kommunikasjonsbærer, for samarbeid og ideutvekslinger underveis, spesielt på programmeringsdelen, i prosjektet. Medstudentene Kjetil, Øyvind og Fabien fortjener også en takk for god innsikt og gode betrakninger både faglig og på livet ellers. Til sist vil jeg takke kjæresten min, Malin, for kor-

returlesing, god støtte og tro på meg gjennom hele studietiden.

Martin Dalbro
Blindern, mai 2008.

Kapittel 2

Innholdsfortegnelse

Innhold

Figurer

Tabeller

Kapittel 3

Sammendrag

Denne oppgaven undersøker om det er mulig å bruke ultralyd som kommunikasjonsbærer i et sensornettverk, og effektforbruket denne kommunikasjonsmetoden medfører. Oppgaven er en del av et større prosjekt støttet av Hydro, der ulike implementasjoner i et sensornettverk, som kan komme oljeselskapet, til gode blir utprøvd. Motivasjonen for oppgaven har vært behovet for en trådløs dataoverføring som kan fungere under vann, da radiobølger og høyfrekvente bølger har en meget høy dempingsfaktor i vann. Akustiske, lavfrekvente bølger har tidligere vist seg å fungere bedre under vann, og blir derfor utprøvd i dette prosjektet.

I prosjektet er det sett på hvordan ultralydkommunikasjon kan implementeres i et laveffekts sensornettverk. Det er brukt sensornoder av typen Tmote Sky produsert av Moteiv i prosjektet. Ved å velge den modulasjonsmetoden som synes best egnet, og deretter tilpasse elektronikken rundt ultralydtransduserne til sensornodene, med de løsningene som synes å være den best fungerende, har konstruksjonen tatt form. Konstruksjonen har deretter blitt bygget og utprøvd med hensyn på effektforbruk og ytelse. Alternative forbedringer av den utprøvde konstruksjonen har deretter blitt foreslått. Parallelt har det blitt sett på hvordan programvaren på Tmote Sky sensornodene kan tilpasses til akustisk dataoverføring.

Kapittel 4

Innledning

4.1 Bakgrunn

I oljeindustrien benyttes det idag kabler for kommunikasjon mellom boreplatform og utstyr på havbunnen. Da kabler kan ryke er det ønskelig å ha en alternativ kommunikasjonskanal for overvåkning og eventuelt også nødstopp mellom oljeplatformen og boreutstyret på havbunnen, men da radiobølger har en høy dempingsfaktor i vann vil konvensjonell radiokommunikasjon fungere dårlig. De store dypene boringen foregår på gir lange avstander mellom oljeplatformene og utstyret på havbunnen, noe som i enda større grad vanskeliggjør bruk av direkte radiokommunikasjon. Ideen er derfor å ta i bruk et trådløst sensornettverk med relefunksjon for å dele opp kommunikasjonsveien i flere deler. Et sensornettverk med relefunksjon har den egenskapen at noder tilknyttet et nettverk kan kommunisere med hverandre via andre noder i nettverket uten å ha direkte radiokontakt.

Det er også et velkjent faktum at dempingsfaktoren i vann øker med høyere frekvens. Tanken er derfor å se på alternative, mer lavfrekvente måter å kommunisere under vann. Det ble bestemt at student Aart Joakim in't Veld skulle gjøre forsøk med infrarødt lys som kommunikasjonsbærer og at undertegnede skulle forsøke med ultralyd, som er lyd med frekvens over det mennesket kan oppfatte, altså over 20kHz. Ultralyd kan legges til en lavere frekvens enn frekvenser ved bruk i radiokommunikasjon, og signalets rekkevidde vil dermed øke. Ultralyd setter også selve kommunikasjonsmediet i bevegelse, i motsetning til radiokommunikasjon som bruker elektromagnetiske bølger. Det at kommunikasjonen skal foregå trådløst impliserer også at nodene er plassert uten tilgang på annen strømforsyning enn separate batteripakker. Da nodene vil være plassert i

vann, vanskelig tilgjengelig for bytte av batteri, bør også strømforbruket holdes lavt.

4.2 Mål/Problemstilling

Målet med denne oppgaven er å prøve ut om det er mulig å få ultralyd til å fungere som kommunikasjonsbærer for digital data over analoge signaler med et sensornettverk. Fokuset i oppgaven vil være ultralyd, der det blir undersøkt om ultralyd som kommunikasjonsbærer i det hele tatt vil fungere, hvilken overføringsrate man eventuelt kan få til, hvor pålitelig systemet blir og hvor mye strøm som trekkes i systemet. Det hele skal dessuten styres med sensornoder fra Moteiv av typen Tmote Sky. Det skal lages kort for ultralydkommunikasjon som kobles til Tmote Sky sensornodene.

4.3 Metoder

Tmote Sky sensornodene som brukes i prosjektet er utviklet for å fungere i et sensornettverk med radiokommunikasjon på kommunkasjonsstandarden IEEE 802.15.4 [33]. Sensornodene blir brukt til å styre trådløs kommunikasjon over ultralyd. Eksterne komponenter, som ultralydtransdusere kan kobles til Tote Sky sensornodene via sensornodenes utbyggingskonnektorer. De to eksterne sender- og mottakermodulene kan da styres med Tmote Skys mikrokontroller, *Texas Instruments MSP430*, som kjører operativsystemet TinyOS 2.0. TinyOS 2.0 er et operativsystem utviklet for laveffekts sensornettverk, og programmeres med programmeringsspråket "nesC", *network embedded systems C*. "nesC" er et C-basert, hendelsesdrevet programmeringsspråk for sensornoder, utviklet ved Berkeley University of California. For å få sender- og mottakertransduserne til å fungere sammen med Tmote Sky sensornodene blir det bygget kort med elektronikk som tilpasser Tmote Sky og ultralydtransduserne til hverandre. Simuleringer av hver enkelt del av sender- og mottakerkortene gjøres i det SPICE-baserte [34] simuleringsprogrammet *SwitcherCAD III* fra *Linear Technology* [22] før den konstruksjonen som virker mest lovende blir fysisk konstruert på printkort og testet ut. All utprøving og dokumentasjon av de fysiske kortene gjøres i Matlab, som styrer og leser av tilgjengelig utstyr som funksjonsgenerator, voltmeter, oscilloskop og strømforsyning. Samtidig skrives det "nesC"-kode til Tmote Sky sender- og mottakerkortene for utprøving av kommunikasjonen mellom sender og mottaker.

4.4 Verdt å merke seg

Senderdelen og mottakerdelen som kobles på Tmote Sky sensornodene er konstruert som to separate kort. De to kortene kan likevel kobles til samme sensornode da de, bortsett fra strøm- og jordtilkoblingene, ikke bruker de samme utgangene på Tmoten til lesing og styring av kortene. Kortene vil da kobles i høyden og fungere som ønsket. Programmene for sending og mottak kan også integreres til en enhet og fungere på et kort. Kortet kan dermed motta data via mottakerkortet og sende data videre via senderkortet, eller det kan settes opp en toveiskommunikasjon. Da dette er et uttestingsprosjekt der prinsippet med ultralydkommunikasjon testes ut har ikke det blitt gjort, men det er lagt til rette for at begge de to kortene skal kunne kobles til samme mottaker. Likevel vil ultralydkommunikasjon ha begrensninger i forhold til radiokommunikasjon med tanke på direktivitet. Dette vil taes opp i kapittel 6, *Diskusjon*.

Kapittel 5

Materialer og metode

Målet med denne oppgaven er å lage et system for ultralydkommunikasjon i sensornettverk. Utgangspunktet for arbeidet var Moteivs Tmote Sky trådløse sensornoder for sensornettverk, og det ble fort klart at det måtte bygges eksterne sender- og mottakerkort. Det var også et ønske om at systemet skulle bruke lite strøm, slik at batterienes levetid ble lang. Da både Moteivs Tmote Sky sensornoder og operativsystemet TinyOS 2.0 har hatt flere begrensninger enn først antatt har det underveis vært behov for endringer av de eksterne kortenes opprinnelige konstruksjon.

Utviklingsprosessen av sender- og mottakerkortene har foregått ved å sette opp forskjellige konstruksjonsmuligheter, beregne komponentstørrelser og teste konstruksjonen av de forskjellige komponentene separat i simulering. Når en konstruksjon har fungert som ønsket ved simulering har konstruksjonen, eller deler av konstruksjonen blitt bygget og testet. Den endelige konstruksjonen har så blitt finjustert ved å endre komponentstørrelser til å fungere nøyaktig som spesifisert. Først vil det i oppgaven bli en overordnet gjennomgang av hvordan konstruksjonen skal fungere, deretter vil hvert element bli gjennomgått i avsnittene *instrumentering*, *Senderkortet* og *Mottakerkortet*. Programmeringen av Tmote Sky sensornodene, med de utfordringer det innebar blir beskrevet som en egen del. Underveis knyttes programmeringen opp mot resten av konstruksjonen og mot teori om programmeringsspråket "nesC", TinyOS og Tmote Sky sensornodene.

5.1 Ønsket funksjonalitet

Det ble tidlig foreslått at *frekvens Shift Keying*, FSK var den mest stabile og robuste måten å kommunisere på med trådløs ultralyd. Valg av

modulasjonsmetode blir tatt opp i avsnittet *modulasjon*. Som en følge av at FSK ble valgt som modulasjonsmetode ble det behov for å generere to ulike frekvenser for å skille mellom logisk 0 eller logisk 1. Den første ideen var da å generere de to frekvensene direkte i Tmote Sky sensornodene på en av de digitale I/O-pinnene omtalt i *Endringer på Tmote Sky*, avsnitt 5.5.1. I/O-pinnen skulle så styre et sendetrinn for ultralydtransduseren. Det viste seg at den valgte ultralydtransduseren ville behøve en spenning på opp mot 30 V for optimal sending av utsignalet, derfor ble det bestemt at sendetrinnet skulle inneholde en standard boostconverter for å pumpe opp spenningen fra 3 V til opp mot 30 V.

Etter å ha sett på løsninger for mottakeren ble det vurdert som mest hensiktsmessig at den skulle bestå av to parallelle filtre etterfulgt av hver sin likeretter og en integrator i form av en kondensator. Når spenningen over en av kondensatorene nådde et valgt nivå, en terskelspenning, skulle det detekteres en mottatt 0 eller 1, avhengig av hvilket filter som hadde sluppet igjennom det mottatte signalet. Spenningen som ble lagret på kondensatorene ble så sammelignet opp mot en valgt spenning på en komparator som utløste en av de to digitale inngangene på Tmote Skys 10-pins utbyggingskonnektor som igjen trigget et "interrupt" eller avbrudd hos Tmote Sky på mottakersiden.

Det ble etterhvert erfart at TinyOS operativsystemet ikke kan levere bedre tidsoppløsning enn ett millisekund [30], noe som gjorde det umulig å generere de to senderfrekvensene direkte i sensornoden. De to sendefrekvensene måtte derfor genereres med hver sin oscillator som ble aktivert med Tmotens digitale utganger. På mottakersiden var det behov for å forsterke opp mottakersignelene etter filtrering, for å få opp rekkevidden på kommunikasjonskanalen og øke forskjellen i spenningsnivå ved mottatt og ikke mottatt signal. En vanlig operasjonsforsterker på hver av de to inngangene ble derfor valgt som forsterker. Da Tmote Sky sensornoden leverer ut 0 V og 3 V spenning ble i og med bruk av operasjonsforsterkere i kretsen behov for å generere signaljord på $\frac{V_{CC}}{2}$ for forsterkerne i kretsen. Dette ble generert ved å koble opp passive komponenter som en spenningsdeler der signaljord måtte genereres.

5.2 Ultralyd- kontra radiokommunikasjon

Mens radiokommunikasjon består av elektromagnetiske bølger, eller elektromagnetisk stråling, består ultralyd av lydbølger som setter mediet bølgene forplantes i i bevegelse. Da det er et velkjent faktum at elektromagnetiske bølger under vann har en meget høy dempingsfaktor[36]

forskes det på å bruke lydbølger som kommunikasjonsbærer ved undervannskommunikasjon, da dempingsfaktoren under vann for lydbølger er lavere enn for elektromagnetiske bølger. Dempingsfaktoren er også avhengig av frekvensen til et signal: Høyfrekvente signaler har en høyere dempingsfaktor enn lavfrekvente signaler [37]. Ved å holde frekvensen lav kan altså dempingsfaktoren gjøres relativt lav. Mens bølgene i radiokommunikasjon beveger seg med en hastighet tilnærmet lysets hastighet, $(3 * 10^8 m/s)$, vil lydbølger under vann bevege seg med en mye lavere hastighet, typisk 1500 m/s med små variasjoner avheningig av trykk, saltinnhold og temperatur. Dette er over 4 ganger så fort som lydens hastighet i luft, men bare 1/200000 av hastigheten til radiobølger. Denne hastighetsforskjellen gjør at kommunikasjonsprotokollene brukt i radiokommunikasjon ikke kan brukes umodifisert i ultralydkommunikasjon.

5.3 Modulasjonsmetoder

Analoge signaler er kontinuerlige signaler som kan beskrives med en matematisk tilnærming som en sum av mange sinussignaler. Et sinussignal kan karakteriseres ved amplitude, frekvens og fase. Dermed er det disse tre variablene vi kan velge å endre på for å kode ulike digitale bit eller bitkombinasjoner. I følge [7] er det tre grunnleggende modulasjonsmetoder tilgjengelig for å kode digitale data med analoge signaler: Amplitude Shift Keying (ASK), Frekvens Shift Keying (FSK) og Fase Shift Keying (PSK). Mer avanserte kodingsteknikker med analoge signaler kombinerer de tre modulasjonsmetodene.

5.3.1 Amplitude Shift Keying

Signalet i *Amplitude Shift Keying* (ASK) består et sinus-signal med fast frekvens og fase, og variabel amplitude. Ved et-bits koding er det vanlig at en logisk tilstand uttrykkes ved at det ikke sendes noe signal, dvs. amplituden er 0, og at den andre logiske tilstanden uttrykkes med at det sendes et signal med amplitude ulik 0, og kalles da gjerne On-Off Keying (OOK). Dette vises i formel 5.1.

$$s(t) = \begin{cases} Asin(2\pi f_c t) & logisk1 \\ 0 & logisk0 \end{cases} \tag{5.1}$$

ASK blir gjerne veldig følsomt for forandringer i mottatt signalstyrke, da en uventet demping av signalet vil kunne gi en potensiell feil,

og egner seg derfor best ved bruk av kabler. Hvert bit er gitt en amplitude. Ved å la ethvert mottatt signal med amplitude over en lav terskelspennig beskrive et bit, og det å ikke motta noe signal beskriver det andre bit'et blir muligheten for feil begrenset, men dette er ansett som en lite effektiv moduleringsteknikk. ASK eller OOK brukes som regel i forbindelse med optiske fibre, men kan likevel være verdt å se nærmere på, da et ultralydsignal vil kunne fungere uten å måtte filtrere inngangssignalet. Dermed blir elektronikken veldig enkel, og potensiell demping av inngangen i inngangsfiltrene vil elimineres. Dessuten kan sendefrekvensen legges nøyaktig på transdusernes senterfrekvens. En stor begrensning med OOK er at man må vite nøyaktig hvor mange bit som skal sendes og hvor lang tid det er mellom hvert sendte bit. Dermed må også sender og mottaker synkroniseres før datatransmisjonen kan starte.

5.3.2 Frekvens Shift Keying

I *Frekvens Shift Keying* (FSK) har man et signal med konstant amplitude og fase, men som varierer signalet mellom to frekvenser der den ene frekvensen beskriver en logisk 1 og den andre frekvensen beskriver en logisk 0. Det utsendte signalet beskrives i 5.2.

$$s(t) = \begin{cases} Asin(2\pi f_1 t) & logisk1 \\ Asin(2\pi f_2 t) & logisk0 \end{cases} \tag{5.2}$$

Det er vanlig at f_1 og f_2 ligger motsatt like langt ifra senterfrekvensen f_c til transduseren som benyttes.

FSK er mindre følsom for feil enn ASK, da hvert bit beskrives av et mer amplitudeuavhengig signal. En viss signalstyrke er naturligvis påkrevd, men ut over dette er FSK mindre følsomt for feil enn ASK. FSK er mye i bruk også i radiotransmisjon, altså i området 3MHz til 30MHz. FSK er dessuten den modulasjonsmetoden som i praksis fungerer i ultralydkommunikasjon [2]. Ved frekvens Shift Keying kan dessuten all signalprosessering gjøres analogt med analoge filtre på inngangen.

5.3.3 Fase Shift Keying

I *Fase Shift Keying* (PSK), brukes det flere faser av et sinussignal til å beskrive de logiske tilstandene. Fasene er da som oftest faseforskjøvet mest mulig i forhold til hverandre. For et signal med to faser vil det innebære at de to signalene er faseforskjøvet 180° (π radianer) i forhold til hverandre. Dette innebærer det samme som å flippe sinuskurven

om signalets DC-verdi som uttrykt i formel 5.3, eller multiplisere det med -1 som uttrykt i formel 5.4. PSK er avhengig av at senderen kan sende ut et pent sinussignal med full kontroll over fasen. Når utsignalet fra transduseren styres av transistorer via et firkantsignal, som i dette prosjektet, vil muligheten for faseforvrengning øke og presise faseskift vil vanskeliggjøres. PSK vil ha større krav til sendetrinnet, og være mindre robust og smidig å jobbe med enn FSK og ASK.

$$s(t) = \begin{cases} Asin(2\pi f_c t) & logisk1 \\ Asin(2\pi f_c t + \pi) & logisk0 \end{cases} \tag{5.3}$$

$$s(t) = \begin{cases} Asin(2\pi f_c t) & logisk1 \\ -Asin(2\pi f_c t) & logisk0 \end{cases} \tag{5.4}$$

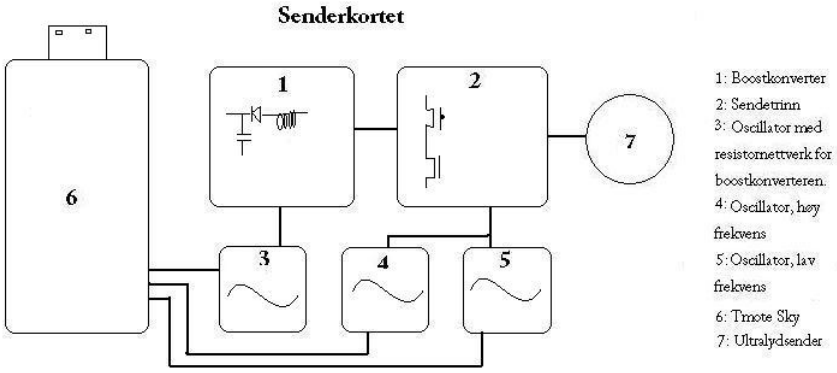
FSK er dermed den modulasjonsmetoden som ser ut til å være mest robust, og derfor blir dette valgt som modulasjonsmetode for ultralydkommunikasjonen i dette prosjektet.

5.4 Oppsett

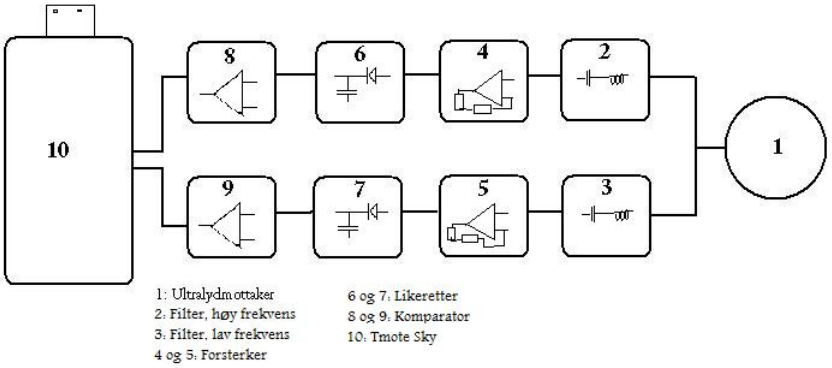
Først og fremst består systemet av to hoveddeler montert på hvert sitt eksterne utbyggingskort som begge blir styrt av Tmote Sky sensornoder. I figur 5.1 og figur 5.2 er det vist med enkle figurer hvordan kortene er bygget opp. Da de to utbyggingskortene ikke bruker samme inn- og utganger på utbyggingskonnektorene kan de to kortene kobles til samme sensornode. Dermed kan hver sensornode ha muligheter for både å sende og å motta signaler, og fungere fullgodt som et bindeledd mellom sensornoder uten direkte kontakt med hverandre, eller fungere med toveiskommunikasjon mellom to sensornoder.

5.4.1 Mottakeren

Mottakeren består av en *Murata MA40S4R* ultralydmottaker, fra nå omtalt som ultralydmottaker, som mottar signaler rundt en senterfrekvens på 40kHz [31], etterfulgt av to parallelle båndpassfiltre som skiller de to senderfrekvensene fra hverandre. Etter hvert filter følger en operasjonsforsterker koblet opp med et motstandsnettverk for forsterkning og generering av signaljord. Etter hver forsterker blir signalet likerettet med en enkel halvperiodelikeretter og lagret på en kondensator koblet til jord. For at kondensatoren skal trekkes ned mot lav spenningsreferanse når den ikke lades opp via halvperiodelikeretteren, er det koblet en motstand til



Figur 5.1: Tegning av senderkortets oppbygging.



Figur 5.2: Tegning av mottakerkortets oppbygging.

jord i parallell med kondensatoren. Kondensatoren er koblet til den positive inngangen på en komparator svitsjer fra lavt til høyt signal når spenningen går over en spenning satt på komparatorens negative inngang. Den stigende flanken trigger et avbrudd hos Tmote Sky sensornoden som leser dette som et mottatt bit. Mottakerkretsen henter 3 V spenning og jord fra Tmote Skys pinner på utbyggingskonnektoren for dette.

5.4.2 Senderen

Senderen i kretsen består av en *Murata MA40S4S* ultralydsender, fra nå omtalt som ultralydsender, som sender signaler rundt en senterfrekvens på 40kHz [32]. Denne senderen er koblet til et sendertrinn for å styre 30 V utspenninga med 3 V styrespenninger. Sendertrinnet består av en P-type transistor og en N-type transistor som styres ved hjelp av relakseringsoscillatorer. Senderkortet har til sammen tre relakseringsoscillatorer: To som styrer hver sin senderfrekvens, som igjen styrer sendertrinnets transistorer, og en som styrer svitsjingen av en boostkonverter via et transistor-resistornettverk for justerbar duty cycle, da utspenningen fra en boostkonverter er avhengig av duty cyclen. Ultralydsenderen krever en spenning på opp mot 20 V topp-til-topp for å fungere godt, og ifølge [2] håndterer den uten problemer spenninger på minst 30 V. En spenning på minst 30 V vil derfor genereres med en standard boostconverter. Boostconverteren er detaljert omtalt i avsnitt 5.6.1 og bruker en N-type transistor i svitsjingen. Senderkretsen henter spenning og jord fra Tmote Skys pinner på utbyggingskonnektoren for dette.

5.5 Instrumentering

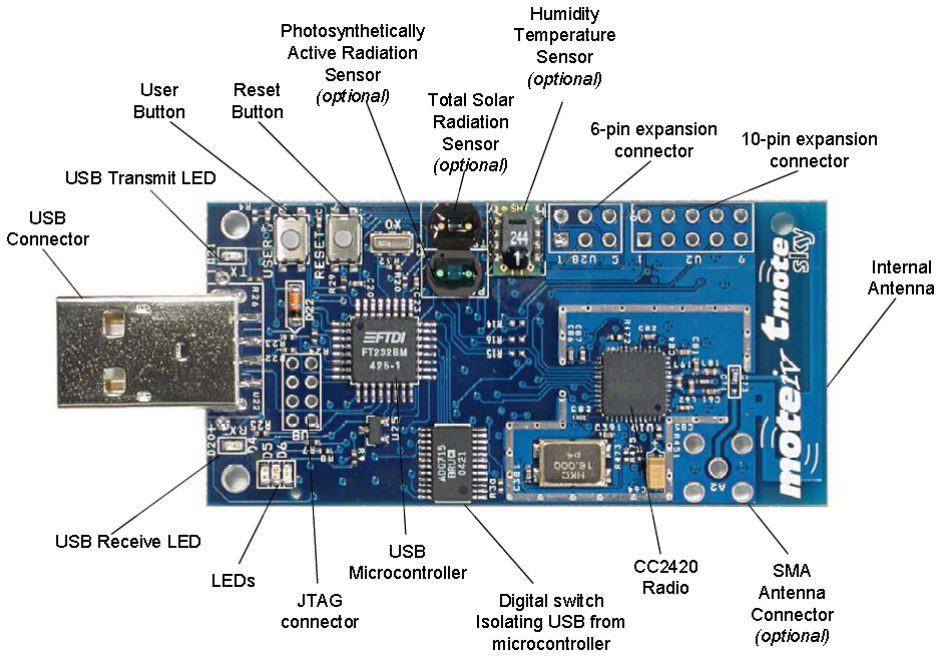
5.5.1 Tmote Sky

Generelt

Tmote Sky er en ultra laveffekts modul for sensornettverk. Den ble opprinnelig navngitt Telos, og er utviklet ved UC Berkeley. En revidert utgave av Telos, kalt TelosB er den sensornoden som nå selges som Tmote Sky. Fra 2005 har den blitt distribuert via Moteiv Corporations, som nå er kjøpt opp av Sentilla [12].

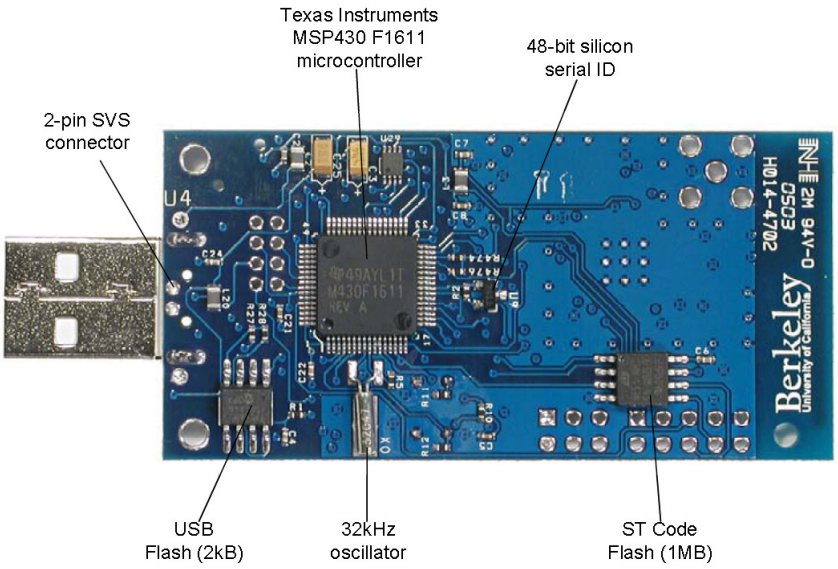
Nøkkelfunksjoner ved Tmote Sky er:

- 250kbps 2.4GHz IEEE 802.15.4 Chipcon CC2420 Trådløs radio.



Figur 5.3: Tmote Sky sensornodens forside.

- Kan samhandle med andre IEEE802.15.4 enheter.
- 8MHz Texas Instruments MSP430 mikrokontroller med 10k RAM og 48k flashminne.
- Integrert ADC, DAC, SVS (Supply Volt Supervisor) og DMA-kontroller.
- Integrert mikrostrip design antenne for radiokommuniksjon, 50m rekkevidde innendørs, 125m rekkevidde utendørs.
- Mulighet for integrerte sensorer for luftfuktighet, lysstråling og temperatur, samt temperatur i prosessoren.
- Meget lavt strømforbruk.
- Rask oppvåkning fra hvilemodus: 6 μ sekunder.
- Programmering og datatransmisjon til pc via USB.
- 10-pins og 6-pins konnektor for eksterne moduler/sensorer.
- Mulighet for SMA antennetilkobling.



Figur 5.4: Tmote Sky sensornodens bakside.

MSP430 mikrokontroller

Texas Instruments MSP430 mikrokontroller bruker meget lite strøm både i aktiv- og i hvilemodus [12]. I aktivmodus er mikrokontrolleren drevet av en intern digital oscillator som kan operere med frekvenser opp til 8MHz. Denne digitalt kontrollerte mikrokontrolleren kan aktiveres fra hvilemodus på bare 6 μ sekunder. I hvilemodus drives mikrokontrolleren av en 32 kHz (32768Hz) krystall.

Chipcon CC2420 radioen

Radioenheten til Tmote Sky er en enkel Chipcon CC2420 chip som er kompatibel med IEEE 802.15.4 2,4 GHz kommunikasjonsstandarden [33]. Den gir en datarate på 250 kbps, og støtter datahåndtering, buffering, og indikerer linkkvaliteten. Den styrer altså alt som har med radiokommunikasjon å gjøre. Den styrer innkommende pakker, og avgjør ut ifra pakkens adresse om den mottatte pakken skal sendes videre til selve MSP430 mikrokontrolleren som TinyOS operativsystemet kjører på. Ideelt sett ville det vært ønskelig å tilpasse IEEE 802.15.4 kommunikasjonsstandarden til å kunne bruke 40kHz ultralydkommunikasjon, men utfordringer med dette er omtalt i avsnitt 5.5.1 om programvare.

Programvare

Til Tmotene følger det med en del ferdig programvare for å kunne teste og lære seg å bruke Tmote Sky. Selve programmeringen foregår i et Linux-lignende skall som kalles *Cygwin* eller direkte i *Linux (Ubuntu)*. I prosjektet brukes det en Linuxvariant som er spesielt tilpasset TinyOS-programmering, kalt *Xubuntos*. I *Cygwin* eller *Linux* kompileres programkode som så lastes over på Tmote Sky sensornoden via den innebygde USB-tilkoblingen. Hver Tmote Sky sensornode blir da gitt en unik adresse, samt satt til å være "master"/koordinator eller slave i systemet. Valg av adresse og om noden skal være koordinator eller slave kan gjøres automatisk eller manuelt ved hjelp av tillegg i opplastningskommandoen.

I og med IEEE 802.15.4 standarden som er implementert i radiokommunikasjonen på Tmote Sky vil to endenoder som ikke har direkte kontakt kunne kommunisere med hverandre via andre noder i systemet. To noder i et sensornettverk kan kommunisere med hverandre uten å ha direkte kontakt, men via andre noder. Dette styres i Chipcon CC2420 radioenheten.

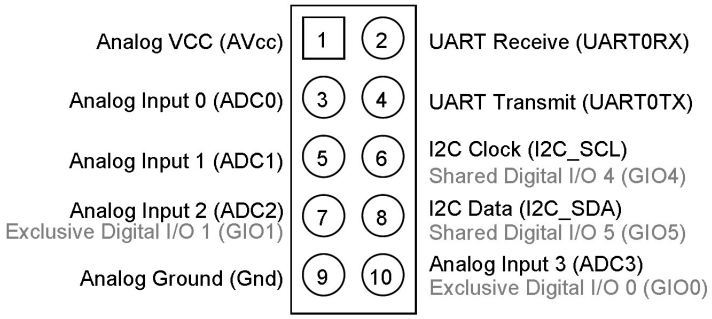
Selv om Tmotene leveres med ferdige programmer for trådløs kommunikasjon vil ikke disse fungere uten videre med ultralyd som kommunikasjonsbærer, da bærefrekvens, signalhastighet og modulasjonsmetode som brukes i radiokommunikasjonen er forskjellig fra ultralydkommunikasjonen som benyttes i dette prosjektet. Opprinnelig blir all radiokommunikasjon dessuten styrt i en *Chipcon CC2420* radiodel [12] via *Texas Instruments* mikrokontrolleren. Radiodelen kan ikke programmeres til å styre utbyggingskonnektorene på Tmote Sky sensornodene og dermed kan de heller ikke brukes i forbindelse med ultralydkommunikasjon. Funksjonen med pakkehåndtering og multihopp blir altså styrt i forbindelse med radiodelen og er en del av IEEE 802.15.4 standarden som ligger implementert i CC2420-chipen. Dermed forsvinner denne funksjonen ved overgang til ultralydkommunikasjon. All kommunikasjonskontroll må da eventuelt styres i operativsystemet TinyOS 2.0 og programmeres i "nesC".

Endringer på Tmote Sky

Tmote Sky sensornodene er modifisert med ny programvare på operativsystemet TinyOS 2.0 for å styre de eksterne sender- og mottakerkortene via utbyggingskonnektorene. Kortene har også fått loddet på 0 Ωs motstander på posisjon R14 og R16, vist i figur 5.5. Motstandene sørger for at I/O-portene, port 7 og port 10 på 10-pinskonnektoren vist i figur 5.6,

fungerer som digitale I/O-porter [9] og ikke analoge innganger.

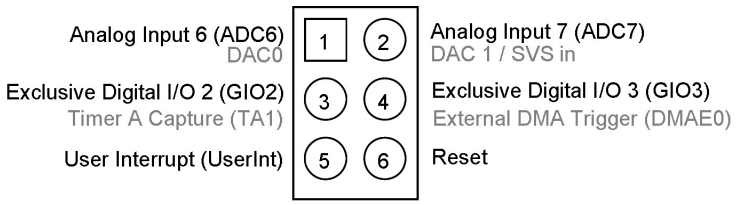
Figur 5.5: Plassering av motstandene R14, R15 og R16



Figur 5.6: Tmote Skys 10-pins utbyggingskonnektor.

Tmote Sky sensornodene har også fått loddet på kontakter på 6- og 10-pinskonnektorene, for å kunne kobles til eksterne sender- og mottakerkortene i prosjektet. Konnektorenes tiltenkte funksjoner er vist i figur 5.6 og i figur 5.7, men det ble erfart at dette kan endres ved å programmere rett på HPL-laget omtalt i avsnitt 5.5.2, altså ved å programmere direkte til mikrokontrollerens registre.

Det viste seg at portene GIO0 og GIO1 som det var tenkt at skulle trigge hver sin relakseringsoscillator ikke ga mer enn 1,25 V spenning ut, noe som var for lavt til å utløse inngangene på de to oscillatorene som ikke utløses før innspenningen er over $\frac{V_{cc}}{2}$, 1,5 V [9]. Derfor ble det gjort forsøk med å bruke Tx- og Rx-portene i figur 5.6 til dette



Figur 5.7: Tmote Skys 6-pins utbyggingskonnektor.

formålet. Utprøving viste at disse portene kunne levere 3 V spenning ut. Da det i dette prosjektet opprinnelig var tenkt at Tx-porten skulle trigge svitjsetransistorens oscillator på senderkortet måtte den funksjonen flyttes til 6-pinskonnektoren. Valget falt da på port 2, analog input-konnektoren siden den, som Tx og Rx-portene, også kunne levere 3 V spenning ut.

Programmering av Tmote Sky

Tmote Sky sensornodene blir programmert med programmeringsspråket ”nesC”, *network embedded systems C*. ”nesC” er utviklet spesielt for sensornettverkssystemer [13], som for eksempel Tmote Sky og er en variant av programmeringsspråket C, med hovedfokus på å være et helhetlig systemkonstruksjons-språk som er sterkt knyttet opp mot maskinvaren. ”nesC” er et statisk språk uten dynamisk minnetildeling, noe som gjør analysen av og optimaliseringen av programmer lettere. Programmeringsspråket er bygget opp i komponentfiler der man lar hver komponentfil gjøre en oppgave. De enkelte komponentfilene knyttes sammen med hverandre og danner tilsammen et helhetlig program. Dermed kan noen ferdigutviklete komponentfiler benyttes i prosjektet, mens andre komponentfiler igjen må skrives spesielt for prosjektet. Alle komponentene i et helhetlig program må til slutt linkes sammen i en konfigurasjonsfil for å fungere som tiltenkt.

Utfordringer og fordeler med nesC

”nesC” er drevet av interaksjon med omgivelsene. I motsetning til tradisjonelle computere brukes sensornoder som regel til å samle inn data og til å kontrollere omgivelsene. Derfor er ”nesC” hendelsesdrevet, og hendelsesankomst og dataprosessering skjer parallelt. Dermed må det prioriteres hva som er viktigst med tanke på tidskrav. Mikrokontrollerne har begrensede data- og minneressurser fordi det stilles strenge krav til størrelse, pris og effektforbruk. Kravene til effektforbruk setter en

naturlig stopper for bruk av kraftige mikroprosessorer med høy regnekraft i sensornoder. En sensornode skal som nevnt gjerne kunne fungere lengst mulig på et lite batteri. Effektforbruk er derfor hovedfokuset i de fleste applikasjoner idag [26].

TinyOS

TinyOS er et operativsystem som er spesielt laget for laveffekts sensornettverk, og må tilfredsstille helt andre krav og behov enn en tradisjonell datamaskin. TinyOS har en hendelsesdrevet programmeringsmodell og krever veldig lite plass, kun 400 byte kode og dataminne for Operativsystemkjernen (OS-kjernen). Da tilgjengelig minne på Tmote Sky sensornodene er meget begrenset er det viktig å kunne lage kompakt kode som gjerne skal kunne brukes flere ganger. Den komponentbaserte arkitekturen i TinyOS gjør at vi ender opp med programkomponenter som kan brukes flere ganger på samme sensornode. Gjenbruken av programkomponenter er minnebesparende, og sammenknytningen av komponenter kan sammenlignes med å knytte sammen prosesser i FPGA-programmering. I TinyOS blir også lavt effektforbruk prioritert i større grad enn for andre operativsystemer, og sanntidskravene er lavere. Dette går ut over tidsoppløsningen og fører til behov for eksterne oscillatorer i dette prosjektet.

TinyOS er utviklet gradvis via Boomerang og TinyOS 1.x, til nyeste versjon, TinyOS 2.0. Forskjellene på TinyOS 1.x og 2.0 er relativt store. TinyOS 2.0 ble gitt ut i slutten av 2006, og har en rekke fordeler i forhold til TinyOS 1.x. Arbeidet i dette prosjektet begynte derfor smått med TinyOS 1.x, men ble underveis byttet ut med TinyOS 2.0, da dette ble tilgjengelig og fordelene med versjon 2.0 ble tydelige. Alt arbeid i TinyOS 1.x ble gjort i Windows, og programmeringen av Tmote Sky sensornodene ble gjort i Cygwin. Da TinyOS 2.0 ble tatt i bruk ble arbeidet flyttet over til det Linuxbaserte operativsystemet *Xubuntos* som ble kjørt via *VMWare* på Windows XP. Med TinyOS 2.0 ble operativsystemet mer stabilt, og enklere å jobbe med.

I midlertid har TinyOS 1.x fordeler i visse applikasjoner. Det viser seg at å enkelt lese av et analogt innsignal kan gjøres på en mye enklere og kjappere måte i TinyOS 1.x enn i TinyOS 2.0. I TinyOS 1.x kan man med en enkel kommando [11] få tilgang til å lese rå data fra de analoge inngangene. I TinyOS 2.0 derimot må tilkoblingen defineres som en egen komponent som må skrives fra bunnen via *"Read"*-grensesnittet [15]. Det alternativet som er vurdert som det mest brukervennlige i TinyOS 2.0 er programmere direkte i mikroprosessorens registre til å trigge

avbrudd på stigende flanke med grensesnittet *HplMsp430InterruptC* i *HPL-laget*. Også TinyOS 1.x har et grensesnitt for dette kalt *Msp430InteruptC*, men det tilsvarende grensesnittet i TinyOS 2.0 er mer brukervennlig og gir flere muligheter. Den ekstra stabiliteten i TinyOS 2.0 gjør det også mer naturlig å bruke dette operativsystemet. For å være sikker på at avbruddsfunksjonen ville trigges som ønsket ble analoge komparatorer konstruert på mottakerens innganger. Mer om komparatoren i del 5.7 om *Mottakerkortet*. TinyOS 2.0 viser seg altså å være det beste alternativet når det skal skrives kode til sensornettverket hvis elektronikken rundt tilpasses til dette.

Koden for sender- og mottakerkortet vises i Appendix, og viser hvordan en teller hos senderen kan sendes over til mottakeren som viser de tre siste bitene i telleren på diodene på Tmote Sky sensornoden. For mer avansert datahåndtering kan koden eventuelt videreutvikles, men dette skulle være fullgodt for å teste ultralydkommunikasjonen via eksternkortene.

5.5.2 Oppbygning av programmeringsspråket

TinyOS er som nevnt bygget opp av flere komponentfiler som kan gjenbrukes. Alle komponentfilene som er i bruk i en programkode linkes sammen av en konfigurasjonsfil (configuration). Hver komponent implementerer definerte grensesnitt for samhandling med andre komponenter, noe som gjør at mindre endringer kan gjøres uten at hele koden må skives om. Det er flere abstraksjonsnivåer mot maskinvaren i TinyOS, og hvert abstraksjonsnivå modelleres som en komponent [35]. Utfordringen med programmeringsspråket "nesC" ligger ikke først og fremst i å skrive komponentfilene, men i å linke disse sammen.

HPL

Hardware Presentation Layer (HPL) presenterer det laveste nivået tilknyttet maskinvaren. Dette lagets oppgave er, som navnet antyder å presentere maskinvarens muligheter. Denne tilknytningen til maskinvaren gir liten frihet i forhold til komponentfilenes oppbygning og implementering. Derfor vil oppbygningen til alle komponenter i HPL ha relativt lik oppbygning. For å integrere HPL-laget på best mulig måte med resten av arkitekturen bør hver komponent inneholde kommandoer for initialisering, start og stopp av maskinvaren den presenterer, "set" og "clr"-kommandoer for registrene som styrer håndteringen, separate kommandoer med beskrivende navn for de mest brukte flaggsettingsoperasjonene,

kommandoer for å aktivere og deaktivere avbrudd fra maskinvaren, og servicerutiner for avbrudd generert av maskinvaren. Det gjøres ingen kontroll over maskinvarens tilgjengelighet i dette laget, derfor må programmering i HPL-laget gjøres med ekstra nøysomhet.

HAL

I det neste laget, *Hardware Adaption Layer* (HAL), legges nyttige abstraksjoner over HPL-laget. Komponentene i dette laget representerer kjernen av arkitekturen. Målet med dette er å skape et grensesnitt som ikke ofrer effektivitet foran bekvemmelighet. Dette grensesnittet eller HPL-grensesnittet bør brukes ved tidskritiske handlinger. Dette laget sørger for at bare en komponent av gangen har tilgang til bestemt hardware. Komponentene i laget skrives spesielt til hver enkelt platform.

HIL

Den høyeste graden av abstraksjon finnes i *Hardware Independent Layer* (HIL) som er uavhengig av hvilken maskinvare som brukes. Dette laget av abstraksjon skal gjøre det enkelt og komfortabelt å skrive programvare for nye maskinvarekomponenter og tar platformspesifikke abstraksjoner fra HAL-laget og gjør dem om til platformuavhengige grensesnitt. Typiske grensesnitt i HIL-laget er programkode for timere og diodelys.

Når det leveres to abstraksjonsnivåer til en maskinvarekomponent betyr det at komponenten kan brukes parallelt på forskjellige abstraksjons nivåer.

5.5.3 Samtidighet

Det å kjøre parallelle prosesser i programvaren er en viktig oppgave i et operativsystem, men det krever mye strøm og regnekraft hos prosessoren operativsystemet kjører på. Både regnekraft og strøm er knappe ressurser på sensornodene, så da TinyOS 2.0 ble konstruert ble parallelle prosesser implementert på en enkel måte: TinyOS har ingen parallelle tråder, og når en oppgave er påbegynt vil hele oppgaven kjøres igjennom før neste oppgave kjøres, såkalt synkron utføring. Det eneste som kan avbryte en oppgave er maskinvarestyrte avbrudd. Når en tråd blir avbrutt kan det føre til uønskede hendelser med tanke på delte variable: Viktige endringer av variablene kan bli avbrutt, og programmet kan i verste fall henge seg opp og måtte startes på nytt. Med sensornoder som ofte er plassert mer eller mindre utilgjengelig kan en omstart være vanskelig å gjennomføre.

I situasjoner der det er viktig at en oppgave kjøres helt igjennom kan man gjøre hele eller deler av oppgaven "atomic". Det innebærer at oppgaven ikke en gang kan avbrytes av hardware interrupt, men kjøres helt igjennom uansett. For ikke å skape store tidsforsinkelser er det viktig at såkalte "atomic" deler av koden gjøres kort, ellers kan tidsforsinkelsen for viktige avbrudd bli lang [14].

I et sensornettverk omhandler dette stort sett radiokommunikasjon og eventuelt annen kommunikasjon, og hendelser som utløses av sensorer og timere. I dette prosjektet kan den begrensede kanalseparasjonen føre til at både den høye og den lave inngangen blir utløst til tross for at bare en av frekvensene sender. Det filteret som har en senterfrekvens som samsvarer med den sendte frekvensen vil slippe igjennom større del av signalet, og derfor er det den inngangen som mest sannsynlig detekterer et høyt signal først. Ved å gjøre de delene av "nesC"-koden som har med bitmottak å gjøre "atomic" vil programvaredelen bli mer robust mot svakheter i elektronikken.

5.5.4 Laveffektsstøtte

Som de fleste mikrokontrollere har også Tmote Skys mikrokontroller støtte for hvilemodus. Tmoten har støtte for fem forskjellige grader av laveffektsmodus, som strekker seg fra å koble ut CPU'en og den interne digitale klokken, til dypeste grad av hvilemodus som skrur av alle timere og bare kan aktiveres av eksterne avbrudd. Tmoten blir satt i dypest mulige grad av hvilemodus hvis den ikke har noen oppgaver som venter på å bli utført. Det er ikke behov for å kode dette i TinyOS da dette gjøres automatisk. Eksterne komponenter derimot må deaktiveres manueltmed kode hvis man ønsker å spare energi her også. En kort beskrivelse av de 5 nivåene av hvilemodus vises i tabell 5.1:

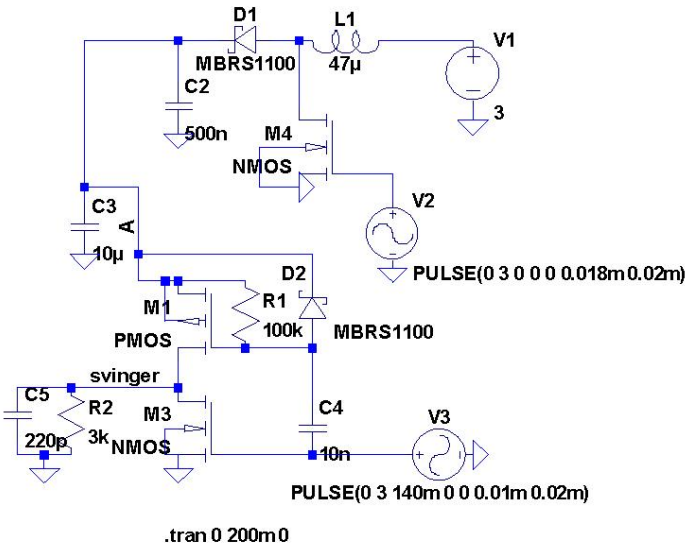
Eventuell strømsparing i dette prosjektet vil altså kun foregå automatisk i Tmote Sky sensornodene når det er mulig. De eksterne kortenes spenningsforsyninger blir altså ikke styrt av Tmotene, og da kortene er rent analoge er det ikke mulig å styre strømforbruket i disse.

Modus	**CPU- og klokkestatus**
`Aktiv`	CPU og alle klokker er aktive
`Modus 0`	CPU og MCLK deaktivert SMCLK og ACLK er aktive
`Modus 1`	CPU, MCLK, DCO er deaktivert DC-generator deaktivert hvis DCO ikke brukes til MCLK eller SMCLK SMCLK og ACLK er aktive
`Modus 2`	CPU, MCLK, SMCLK, DCO er deaktivert DC-generator aktiv ACLK er aktiv
`Modus 3`	CPU, MCLK, SMCLK, DCO er deaktivert DC-generator deaktivert ACLK er aktiv
`Modus 4`	CPU og alle klokker deaktivert

Tabell 5.1: Oversikt over hvilemodus, MSP430 mikrokontroller

5.6 Senderkortet

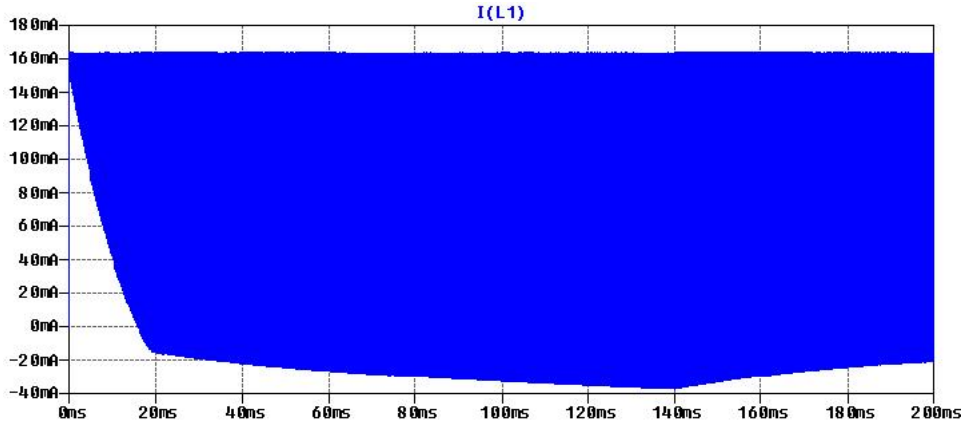
Senderkortet er konstruert med utgangspunkt i Professor Sverre Holms krets for en boostconverter og et sendetrinn drevet av et lavspennings styresignal. Denne konstruksjonen er videreutviklet med styretransistorer som åpnes og lukkes med lave basespenninger, og med oscillatorkretser for generering av sendefrekvenser og styring av svitsjetransistorer. En enkel skisse over konstruksjonen er vist i figur 5.1, og komponentliste er satt opp i Appendix, avsnitt 9.1.1. Boostconverteren med sendetrinnet vises i figur 5.8.



Figur 5.8: Viser boostkretsen med sendetrinn.

5.6.1 Boostkonverteren

Boostconverteren er bygget opp etter helt standard metode for boostconvertere med en spole som er koblet via en svitsjetransistor til jord [38, 2]. Denne koblingen er selve kjernen i boostconverteren. Som svitsjetransistor blir det brukt en vanlig N-type transistor som styres med en egen oscillator. Bryteren vil periodisk være av og på, der tiden den er av lagt sammen med tiden den er på er periodetiden. Dioden vil også oppføre seg som en bryter, som sperrer når styretransistoren er åpen, og omvendt. På katodesiden av dioden er det plassert en kondensator til spenningslagring. I denne applikasjonen er det ønske om minst 30 V utspenning fra boostconverteren for å få høyest mulig signal ut fra ultralydsenderen.



Figur 5.9: Strømmen igjennom spolen i boostkonverteren.

Når bryter M4 i figur 5.6 er skrudd på får vi en enkel krets bestående av en spole koblet til jord via bryteren, fordi dioden da sperrer. All spenningen ligger over spolen, og er koblet til jord via svitsjetransistoren. Spenningskildens spenning legger seg dermed over spolen, og strømmen igjennom denne øker lineært med tiden bryteren er skrudd på. Strømmen igjennom spolen i det bryteren blir skrudd av kan uttrykkes matematisk med formel 5.5.

$$i_L(T_{on}) = i_L(0^+) + \int_0^{T_{on}} \frac{V_s}{L} dt = i_L(0^+) \frac{V_s}{L} T_{on} \tag{5.5}$$

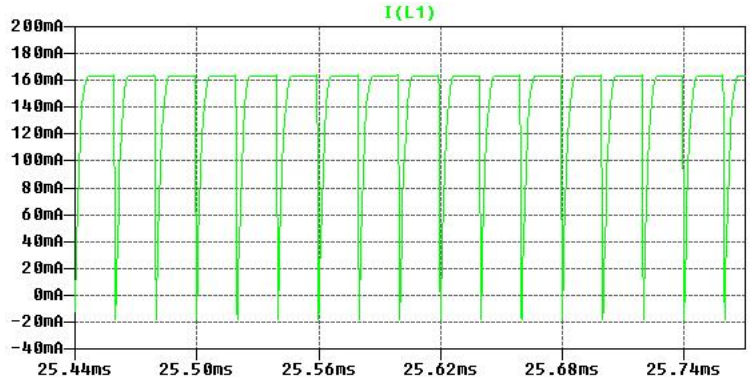
der $i_L(0^+)$ er strømmen når styretransistoren åpnes, V_s er kildespenningen, V_o er utspenningen, T_{on} er tiden styretransistoren er åpen, og T_{off} er styretransistoren er stengt.

Strømmen gjennom spolen kan altså uttrykkes som i formel 5.6 der V_{in} er forsyningsspenningen.

$$\frac{di}{dt} = \frac{V_{in}}{L} \tag{5.6}$$

Det simulerte strømtrekket er vist i figur 5.9 der spolen det refereres til er spole L1 i figur 5.8. Strømmen gjennom spolen varierer mellom 160 mA og -38 mA med samme frekvens som svitsjetransistor N4 i figur 5.8 svitsjer med. I figur 5.10 er det zoomet inn på en kort tidsperiode som viser hvordan strømmen endrer seg, og ikke bare et stort blått felt.

Når svitsjetransistor M4 åpnes vil strømmen bare kunne gå igjennom dioden. Spenningen som settes opp av spolen vil derfor øke til spenningen på katodesiden av dioden + et diodedropp. Denne spenningen lagres på kondensatoren på katodesiden av dioden og kan så brukes til å drive



Figur 5.10: Et kort tidsutsnitt av strømmen igjennom boostkonverterens spole.

ultralydsenderen med en høyere spenning enn forsyningsspenningen på 3 V via sendetrinnet. Kondensatoren lades opp og spenningen stiger som uttrykt i formel 5.7.

$$\frac{dV}{dt} = \frac{i}{C} \tag{5.7}$$

der i er strømmen gjennom spolen i det bryter M4 åpnes.

Strømmen gjennom spolen når svitsjetransistoren blir skrudd på igjen kan uttrykkes med formel 5.8.

$$i_L(T) = i_L(T_{on}) + \int_{t_{on}}^{T_{on}+T_{off}} \frac{V_s - Vo}{L} dt = i_L(0^+)\frac{V_s}{L}T_{on} + \frac{V_s - Vo}{L}T_{off} \tag{5.8}$$

Formel 5.9 gir utspenningen som funksjon av duty-cycle (δ) og innspenning V_s.

$$V_{ut} = \frac{V_s}{1-\delta} \tag{5.9}$$

Ved å snu på formel 5.9 får man formel 5.10 som uttrykker hvilken duty cycle som gir ønsket utspenning.

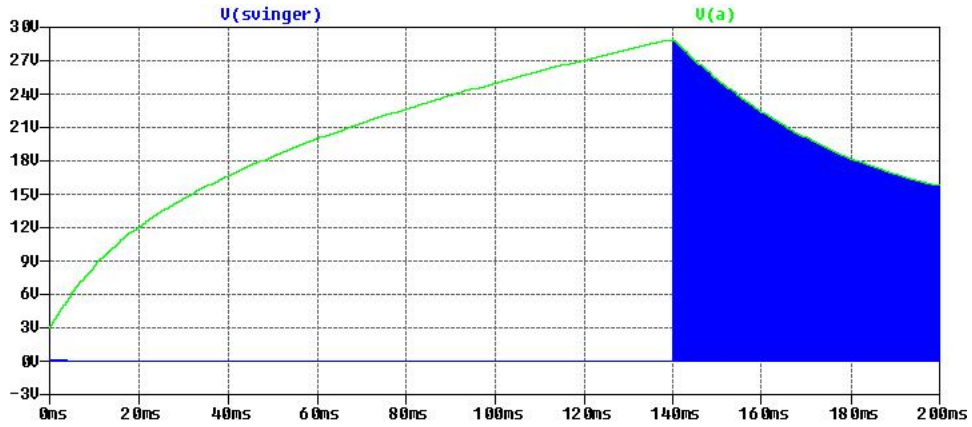
$$\delta = 1 - \frac{V_s}{V_{ut}} \tag{5.10}$$

I dette prosjektet er det ønskelig å hente ut 30 V fra en 3 V spenningskilde. Dette settes inn i formel 5.10, og utregningen vises i formel 5.11.

$$\delta = 1 - \frac{V_s}{V_{ut}} = 1 - \frac{3}{30} = 0.9 \tag{5.11}$$

Det kreves altså ha en duty-cycle på 0.9 for å hente ut 30V spenning fra boostconverteren ved 3V forsyningsspenning. Transistorsvitsjen styres av en oscillator som gir en duty-cycle på ca. 0.9. Konstruksjonen av denne er beskrevet i avsnitt 5.6.3 om *Frekvensvelgere*.

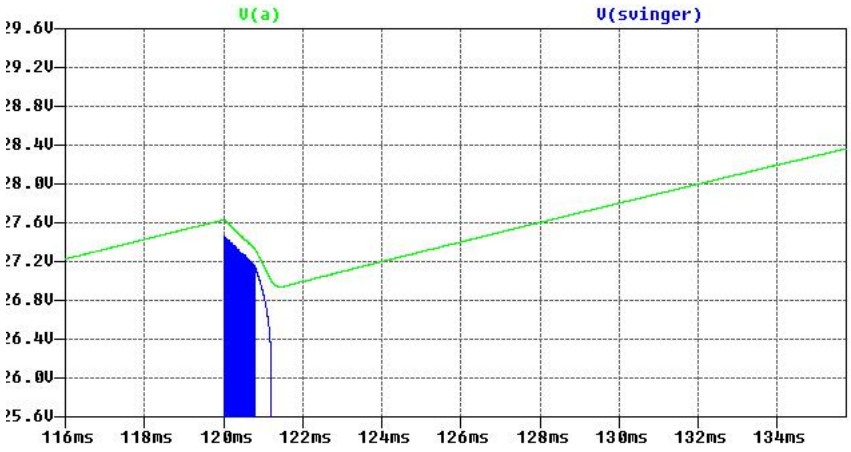
Simulering av kretsen ga resultatet i figur 5.11. Figuren viser hvordan spenningen ut fra boostkonverteren vil øke når man ikke trekker noen spenning i kretsen, for så å aktivere sendertrinnet og trekke en last tilsvarende ultralydsenderens last. Duty-cyclen er på 0.9, som beregnet i formel 5.11. Spenningen vil altså stige mot 30 V, og deretter synke hvis man sender kontinuerlig. I figur 5.12 vises simulert utspenning når senderen sender ut et signal med varighet 1 ms. I figur 5.13 vises spenningsfallet ved en kontinuerlig sendeperiode på ca 5 ms. De to figurene viser at spenningen faller om lag 3 ganger så fort som den lades opp. Det betyr at det kan sendes 1/4, eller 25% av tiden og fortsatt holde utspenningen høy nok.



Figur 5.11: Spenningen ut fra boostkonverteren.

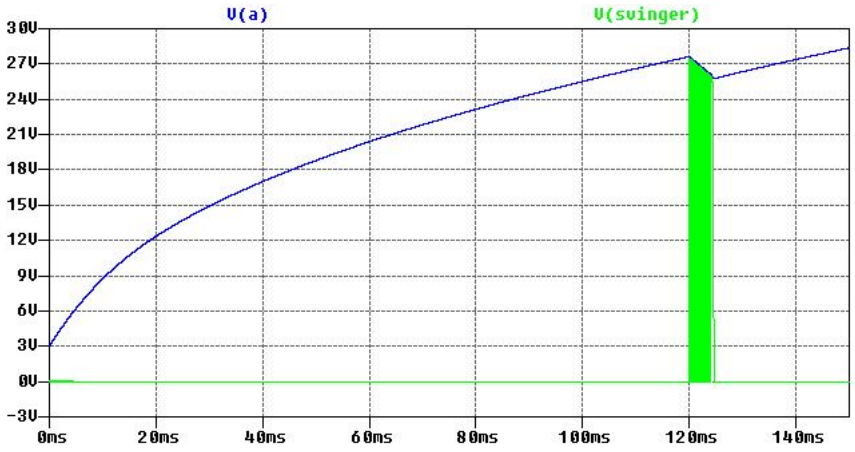
Kondensatoren på diodens katodeside i boostkretsen påvirker oppladningstiden. Stor kondensator gir lengre oppladningstid, men vil som en positiv bieffekt gi mer effektoverskudd til svingeren, da dermed kan sende over lengre tid av gangen. I figur 5.11 vises simulert oppladningstid og simulert fall i spenning ved sending. Ut ifra dette kan kondensatorens størrelse justeres til å være et kompromiss mellom oppladningstid og behov for tilgjengelig energi.

Størrelsen på spolen vil også påvirke strømmen i kretsen og utspenningen. Større spole vil gi en lavere stigning i strøm gjennom spolen, men siden spolen er større vil den også sette opp et kraftigere magnetfelt, noe som gir høyere utspenning. Spisstrømmen fra batteriet vil også bli lavere



Figur 5.12: Spenningsfall over lagringskondensatoren ved 1ms sendeperiode.

med en større spole. Stor spole vil ha høyere seriemotstand enn en mindre spole og begrense strømmen gjennom spolen og dermed effekten av svitsjingen. I følge [38] har boostkonvertere typisk en virkningsgrad på over 0.9. I praksis viser det seg i midlertid at boostkonverteren vil trekke en høy strøm til jord.



Figur 5.13: Spenningsfall over lagringskondensatoren ved 5 ms sendeperiode.

Justering av boostkonverteren

Boostkonverteren ble bygget på et kretskort sammen med de andre delene av senderkortet og testet ut. Utspenning og strømtrekk ble målt med tilgjengelige måleapparater. Ved første forsøk, der alle komponentstørrelser var justert inn etter de teoretiske beregningene, genererte boostkonverteren en spenning på 60 V. Senderkortets totale strømtrekk ble målt til å være 1.25 A. Før boostkonverterens svitsjetransistor ble aktivert trakk senderkortet 6mA strøm. Boostkonverteren trakk med andre ord 1.25 A - 6 mA = 1.244 A. Dette strømtrekket ga altså et effektforbruk på 1244 mA * 3 V = 3732 mW, ikke akkurat laveffekt. Ved å studere de matematiske formlene for strømmen igjennom svitsjetransistoren ser man at strømmen igjennom svitsjetransistor M4 øker med tiden transitoren er åpen. Høyere svitsjefrekvens fører til at transistoren er åpen kortere tid, dermed skulle høyere frekvens bety lavere strømtrekk, noe som bekreftes av formel 5.5. For å begrense strømtrekket til jord ble svitsjefrekvensen økt. Svitsjefrekvensen ble gradvis justert opp til 135,8 kHz, og strømforbruket ble lavere jo høyere frekvensen ble justert. Ved svitsjefrekvens høyere enn 135,8 kHz klarte ikke boostkonverteren å generere høyere spenning enn forsyningsspenningen. Da formel 5.9 viser at utspenningen øker med høy duty-cycle på svitsjetransistor M4, ble duty-cyclen justert ned for å senke utspenningen fra boostkonverteren til i nærheten av 30 V. Etter at boostkonverteren var justert inn ble duty-cyclen beregnet ut fra periodetiden

med formel 5.12.

$$duty = \frac{T_{on}}{T_{total}} = \frac{4,28\mu s}{7,36\mu s} = 0,58 \tag{5.12}$$

Dette stemmer dårlig med de teoretiske beregningene og simuleringene, og kan komme av en høyere spiss-strøm i spolen enn beregnet. Svitsjefrekvensen er også høyere enn beregnet, for å holde strømtrekket nede, noe som indikerer en høyere spiss-strøm i spolen enn ved simuleringene.

Etter at kretsen var justert inn ble strømtrekket målt på nytt, både med og uten sendetrinnet som last. Uten last ble det totale strømtrekket målt til 119mA. Boostkonverterens strømtrekk beregnes da til 119mA - 6mA = 113mA. Når boostkonverteren ble belastet ble det totale strømtrekket målt til 125 mA. Strømtrekket via boostkonverteren var dermed 125 mA - 6mA = 119 mA. Effektforbruket blir da 119 mA * 3 V = 358 mW.

5.6.2 Styrekrets for svinger

Selve svingerens styrekrets er vist i den nederste delen av figur 5.8. Den består av en P-type transisitor over en N-type transistor, som så er koblet til jord. Ultralydsenderen er koblet til mellom P-type og N-type transistoren. De to transistorene styres av en frekvensvelger, som kan gi ut en av de to sendefrekvensene. Dioden som står mellom basen på P-type transistoren og punkt A på figur 5.8 sørger for at basespenningen til transitor M1 i figur 5.8 aldri blir høyere enn spenningen på transistorens kollektor. Motstanden som står i parallell med denne dioden sørger for at basen ikke blir et såkalt flytende punkt der man ikke har kontroll over spenningen. Kondensatoren i parallell med N-type transistoren slipper igjennom styresignalet som styrer P-type transistoren [2]. Sendetrinnet fungerer altså på samme måte som en inverterende krets. Når styrespenningen er lav vil P-type transistoren, som er øverst trekke utgangens spenning til forsyningsspenning (V_{cc}), samtidig som N-type transistoren sperrer jordtilkoblingen. Når styrespenningen er høy vil P-type transistoren sperre for forsyningsspenningen, mens N-type transistoren vil trekke utgangens spenning til jord.

Det ble gjort forsøk med MOSFET-transistorer på styrekretsen, men disse viste seg å ha for høy terskelspenning til å fungere. Normalt vil en MOSFET ha en terskelspenning på typisk 0.7 V, men i og med at det jobbes med så høye spenninger som 30V måtte transistoren dimensjoneres deretter. De tilgjengelige MOSFET-transistorene som tålte slike effekter åpnet ikke før ved rundt 5 V [17] [18] på gatespenningen og kunne dermed ikke brukes i dette prosjektet, som opererer med 3 V

styrespenning. Løsningen ble å bruke bipolare transistorer, da disse kan takle høyere spenning enn MOSFET-transistorer og fortsatt åpne ved en lavere spenning på basen [19] [20], typisk 1,4 V.

5.6.3 Frekvensvelgere

Som beskrevet i avsnittet om operativsystemet *TinyOS* kan ikke TinyOS gi bedre tidsoppløsning enn 1 ms nøyaktighet. Sendefrekvensene må derfor genereres eksternt på utbyggingskortene. Tre alternative måter å gjøre dette på er å bruke en egen mikrokontroller som får dumpet større mengder data før sending, bruke dedikerte timer IC-er, eller det kan gjøres analogt med to oscillatorer, en for hver sendefrekvens. Da en enkel oscillator, og dedikerte timer IC-er fungerer rent analogt, og ikke må programmeres på noen som helst måte er disse løsningene å foretrekke framfor å bruke en egen mikrokontroller.

Relakseringsoscillator

En relakseringsoscillator vil være godt egnet til jobben og er bygget opp med en NAND-gate av typen Schmitt trigger, en kondensator og en motstand som vist i figur 5.6.3 [9]. Nand-gaten bør være av typen Schmitt trigger, da denne har en hysterese på inngangen. Det betyr at triggerspenningene for høy og lav inngang er forskjellig fra hverandre, og muliggjør et visst spenningssving uten at ukontrollerte svingninger oppstår. På den måten unngås det at kretsen svinger når den ikke skal svinge, eller at den svinger med feil frekvens. En relakseringsoscillator fungerer som følgende: NAND-gatens ene inngang er koblet til en av utgangene på Tmoten. Denne styrer om det skal sendes et bit på frekvensen eller ikke. Den andre inngangen er koblet til en kondensator som går til jord, og via en motstand til NAND-gatens utgang.

NAND-gatens utgang svinger da på inngangen for svingerens styrekrets via en inverter. Først ble det vurdert å koble opp de to frekvensvelgerne som vist i figur 5.14, men ved å studere hvordan invertertrinnet er bygget opp ble det fort klart at denne oppkoblingen ville skape en kortslutning mellom de to frekvensvelgerne. Dermed ble oppkoblingen som er vist i figur 5.15 valgt. Utgangen fra de to NAND-gatene ble via hver sin kondensator koblet til en felles inverter som driver sendetrinnet bestående av P-type og N-type transistoren. Det vil da oppstå en spenningsdeling over de to kondensatorene på NAND-gatenes utgang, men spenningen ut til sendetrinnet trekkes opp til fullt spenningssving mellom

0 V og 3 V igjen med inverteren så lenge spenningssvinget på inverterens inngang er stort nok til at den vil invertere innsignalet.

555 timer IC

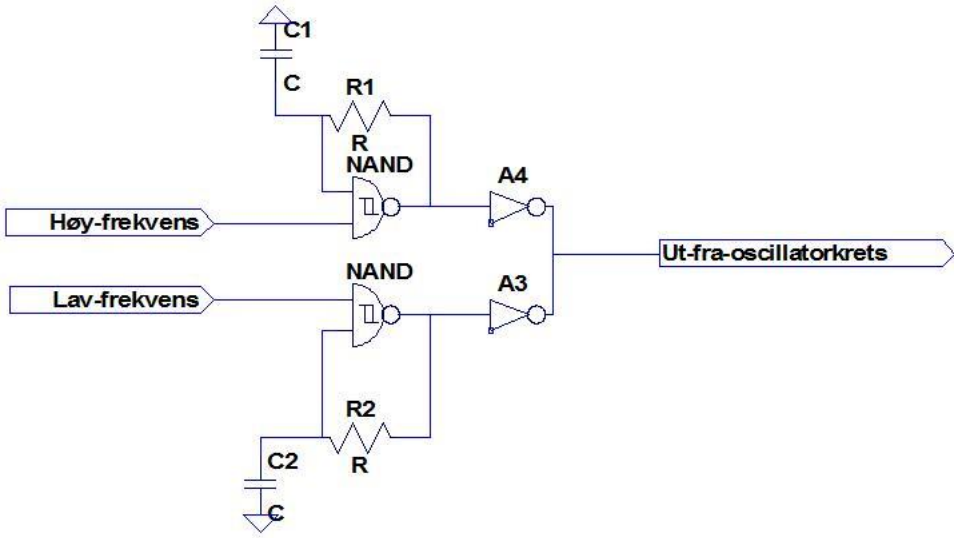
En typisk dedikert timer IC av typen 555 timer er spesielt konstruert for å kunne svinge med en fast frekvens som følge av et RC-ledd [16, 27]. Denne dedikerte timeren kan med en ekstra motstand kobles opp for å gi en høyere duty-cycle enn 0,5, og den garanterer maksimalt 1% avvik fra oscillasjonsfrekvensen. 555 timer IC'en er derimot beregnet på konstant oscillasjon. Ved å drive RC-leddet til hver IC direkte med Tmote Skys styrepinner på utbyggingskonnektorene kan oscillatoren oscillere når et styresignal fra Tmote Sky settes til 3 V. Da to 555 timer-oscillatorer er koblet i parallell på samme måte som med relakseringsoscillatorene og har en inverter på utgangen [27], vil de to inverterne, som da står koblet i parallell, måtte skilles ifra hverandre med kondensatorer og deretter svinges opp til fullt spenningssving igjen med en inverter eller et buffer. En 555 timer er typisk implementert på en 8-pins IC, eller under navnet 556 med to timere på en 14-pins IC. En noe modifisert utgave finnes også under navnet 558 som kombinerer fire 555 timere på en 16-pins IC. Selv om 555 timer IC'en er spesielt utviklet for oscillasjon gjør de begrensede mulighetene for diskontinuerlig oscillasjon at relakseringsoscillatoren virker som et mer brukervennlig alternativ i dette prosjektet. Løsningen med relakseringsoscillatoren er derfor den løsningen som er brukt i dette prosjektet, da denne gir enklere kontrollerbarhet av oscillatorene.

Valgt løsning, Relakseringsoscillator

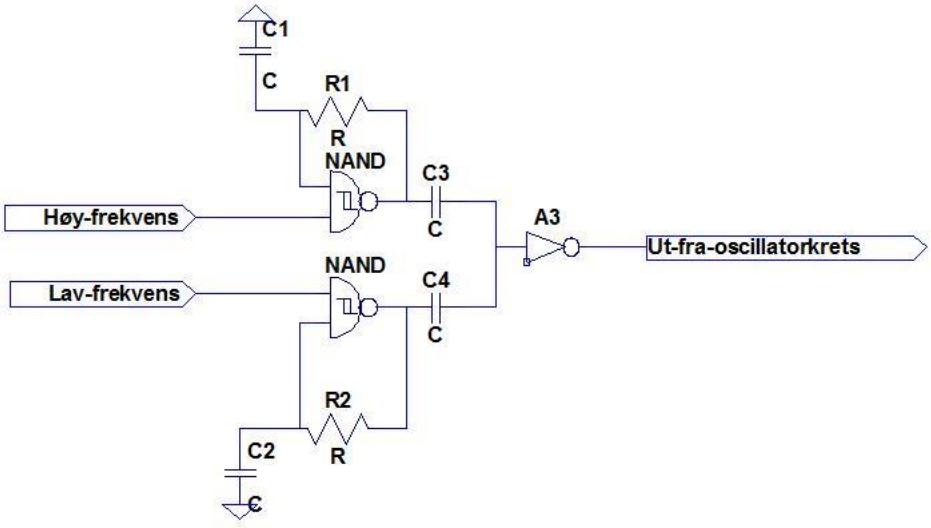
Relakseringsoscillatoren vist i figur 5.6.3 vil svinge som følge av RC-leddet som er koblet til NAND-gatens inngang. Svingefrekvensen finnes ved å beregne tidskonstanten til RC-leddet. Når NAND-gatens utgang går høy vil kondensatoren lades opp via motstanden. Når NAND-gatens øvre terskelspenning er nådd vil utgangen gå lav, og kondensatoren lades ut via motstanden igjen til nedre terskelspenning på inngangen er nådd. Da vil utgangen igjen gå høy, og vi får det samme scenariet. Størrelsen på kondensatoren og motstanden gir opp- og utladningstid, og beregnes ut ifra formel 5.13 [9].

$$f_{osc} = \frac{1}{0.8 * R * C} \tag{5.13}$$

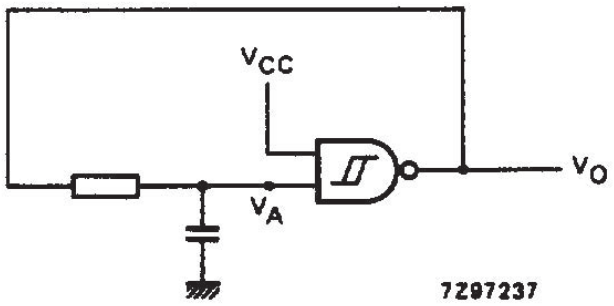
En kondensator på 10nF er valgt, da denne er lett tilgjengelig, og gjør



Figur 5.14: Første frekvensvelgeroppkobling.



Figur 5.15: Frekvensvelgeroppkobling med DC-isolerende kondensatorer.



Figur 5.16: Kretsskjema for relakseringsocillator [9]

om uttrykk 5.13 for å beregne størrelsen på kretsens motstand i formel 5.14.

$$R = \frac{1}{0.8 * C * f_{osc}} \tag{5.14}$$

For 41.8 kHz:

$$R = \frac{1}{0.8 * 10n * 41.8k} = 2,990k\Omega \tag{5.15}$$

For 39.8 kHz:

$$R = \frac{1}{0.8 * 10n * 39.8k} = 3,141k\Omega \tag{5.16}$$

Da de to senderfrekvensene måtte endres til 39,0 kHz og 40,3 kHz er motstandene beregnet på nytt for de nye frekvensene.

For 40.3 kHz:

$$R = \frac{1}{0.8 * 10n * 40.3k} = 3,102k\Omega \tag{5.17}$$

For 39.0 kHz:

$$R = \frac{1}{0.8 * 10n * 39.0k} = 3,205k\Omega \tag{5.18}$$

De to potensiometrene ble justert til rett motstand og satt inn i kretsen. Deretter ble hver oscillator forsynt med 3V spenning og koblet til et oscilloskop for finjustering av frekvensen. Etter finjusteringen ble motstanden i de to potensiometerene målt på nytt. Beregnet motstand, målt motstand og avvik er vist i tabell 5.6.3:

Boostconverterens oscillator måtte bygges opp med et mer avansert system for opp og utladning, da denne gir en dutycycle på opp mot

	Beregnet motstand	Målt motstand	avvik
41.8kHz	2,990kΩ	2,199kΩ	-791Ω
39.8kHz	3,141kΩ	2,265kΩ	-876Ω
40.3kHz	3,102kΩ	2,248kΩ	-854Ω
39.0kHz	3,205kΩ	2,316kΩ	-889Ω

Tabell 5.2: Beregnet og målt motstand, samt avvik for oscillatorene.

0.9. Denne ble da konstruert med separate motstander for opp- og utladning. Opp- og utladningsmotstandene blir valgt via en N-type og en P-type transistor koblet i serie med hver sin motstand, med gaten koblet til Schmitt-triggerens utgang som vist i figur 5.17. Dermed vil N-type transistoren være åpen mens P-type transistoren er lukket, og omvendt. Schmitt-triggerens signal inverteres via en inverter som igjen styrer boostkonverterens svitsjetransistor. Dette betyr at oscillatoren må lages med en duty-cycle på ca 0,1. det vil si at tiden Schmitt-triggerens utgang skal være høy vil være 1/10 av periodetiden, og tiden Schmitt-triggerens utgang skal være lav vil være 9/10 av periodetiden. Når man vet dette kan formelen for motstanden brukes, ved å modifisere periodetiden til å være 0,2 av periodetiden ved en dutycycle på 0,5 for utladningsmotstanden, og til å være 1,8 gang periodetiden for oppladningsmotstanden. Med disse periodetidene kan størrelsen på opp- og utladningsmotstandene beregnes med det samme uttrykket som tidligere. Velger en svitsjefrekvens på 10kHz.

Oppladningsmotstand:

$$R = \frac{1}{0,8 * 10n * (10k * 1,8)} = 6,944k\Omega \tag{5.19}$$

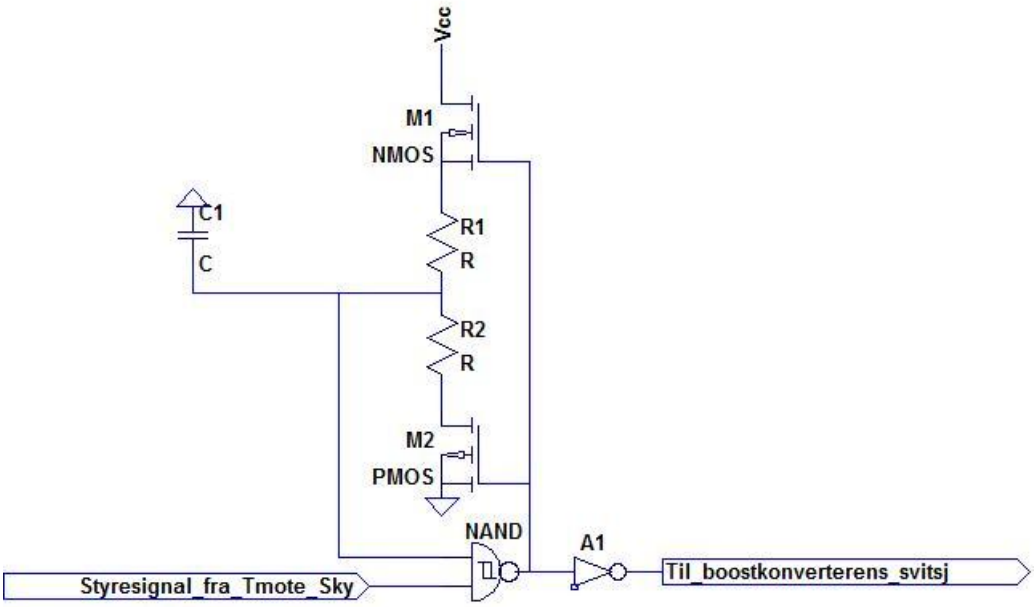
For utladningsmotstanden

$$R = \frac{1}{0,8 * 10n * (10k * 0,2)} = 62,5k\Omega \tag{5.20}$$

Oppladningsmotstanden settes i serie med N-type transistoren, og utladningsmotstanden settes i serie med P-type transistoren. Dette bør gi en duty-cycle på 0,1 på Schmitt-triggerens utgang, som igjen gir gi 0,9 i duty cycle på inverterens utgang og transistorens inngang.

5.6.4 Svingeren

Svingeren i kretsen er en *Murata MA40S4S*, ultralydsender, på sendesiden, og en *Murata MA40S4R*, ultralydmottaker, på mottakersiden. Både ultra-

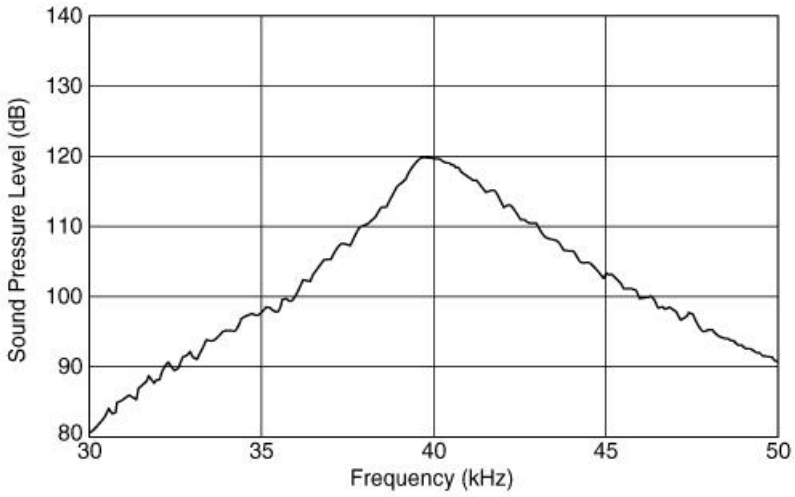


Figur 5.17: Boostkonverterens oscillatorkrets.

lydsenderen og ultralydmottakeren er oppgitt til å ha en senterfrekvens på 40kHz [31, 32]. I figur 5.18 vises ultralydsenderens frekvensrespons [32] oppgitt av produsenten, mens ultralydmottakerens frekvensrespons oppgitt av produsenten vises i figur 5.19 [31].

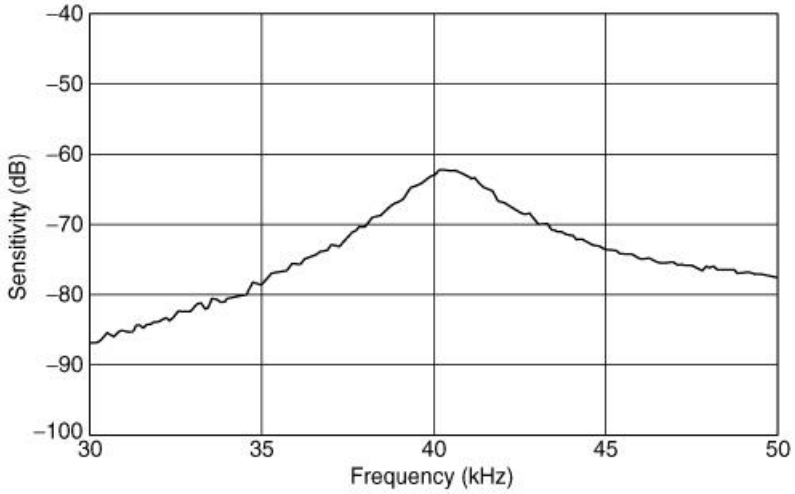
Senderens og mottakerens direktivitet oppgitt av produsenten er vist i figur 5.20 [32] og i figur 5.21 [31].

Ultralydsenderen og ultralydmottakeren ble montert opp rettet mot hverandre med 1,0 cm mellomrom, og senderen fikk sveipet et 10 V topp-til-topp signal fra 30kHz til 50kHz for å kontrollere frekvensresponsen opp mot frekvensresponsen oppgitt i databladene [31, 32]. I figur 5.22 vises den målte frekvensresponsen når transduserne ikke belastes. Den oppgitte frekvensresponsen er altså sammenlignbar med den målte frekvensresponsen, og svingeren skal kunne brukes som antatt.

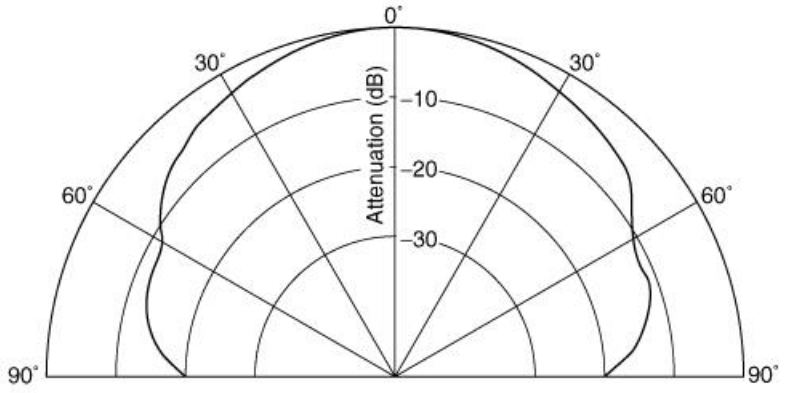


Figur 5.18: Frekvensrespons for ultralydsenderen. [32]

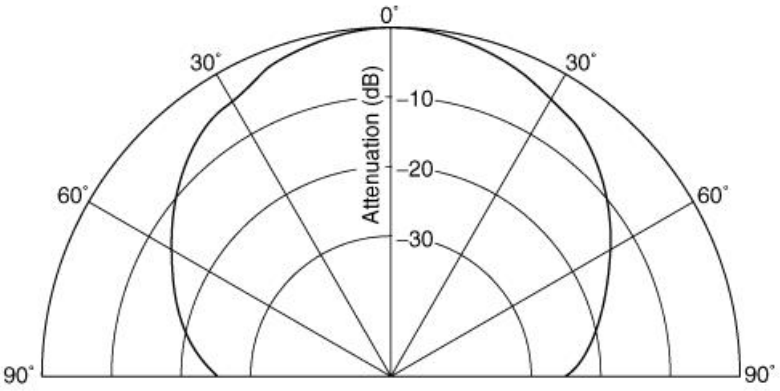
For å kontrollere hvordan mottakertransduseren reagerte på å bli belastet med de to serielle LC-filtrene ble det gjort en måling av frekvensresponsen på samme måte som ved testing av frekvensresponsen uten tilkoblet last. Innsignalet på mottakeren med last tilkoblet ble deretter målt for frekvenser fra 30kHz til 50kHz, som vist i figur 5.23. Man kan se at signalet fra mottakeren faller betraktelig ved de to filtrenes passbånd. Dette kommer av at filtrene ved resonnans får en lav motstand. Dermed klarer ikke ultralydmottakeren å holde signalsvinget på utgangen like høyt som uten last. En måte å motvirke fallet i utspenning på kan være å øke inngangsmotstanden og utgangsmotstanden til filtrene, men med et serielt LC-filter vil, som beskrevet i 5.7.3 økt motstand gi et mindre skarpt filter. Dermed må størrelsen til inngangsmotstanden og utgangsmotstanden i det serielle LC-filteret velges som et kompromiss mellom skarphet i filteret og demping av mottakertransduserens innsignal.



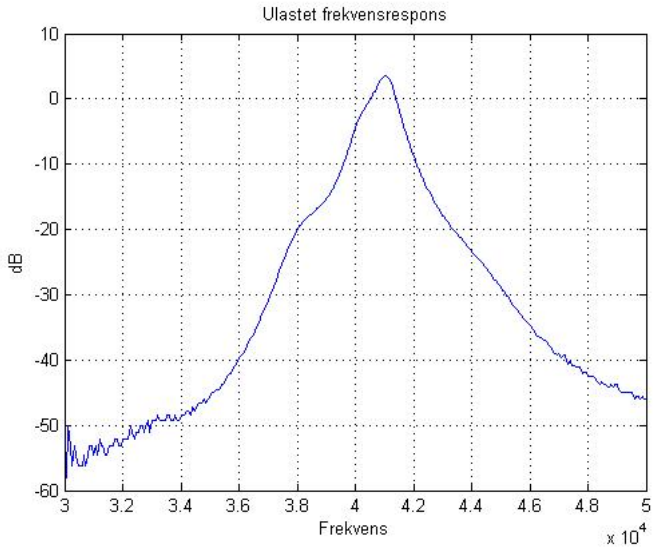
Figur 5.19: Frevensrespons for ultralydmottakeren. [31]



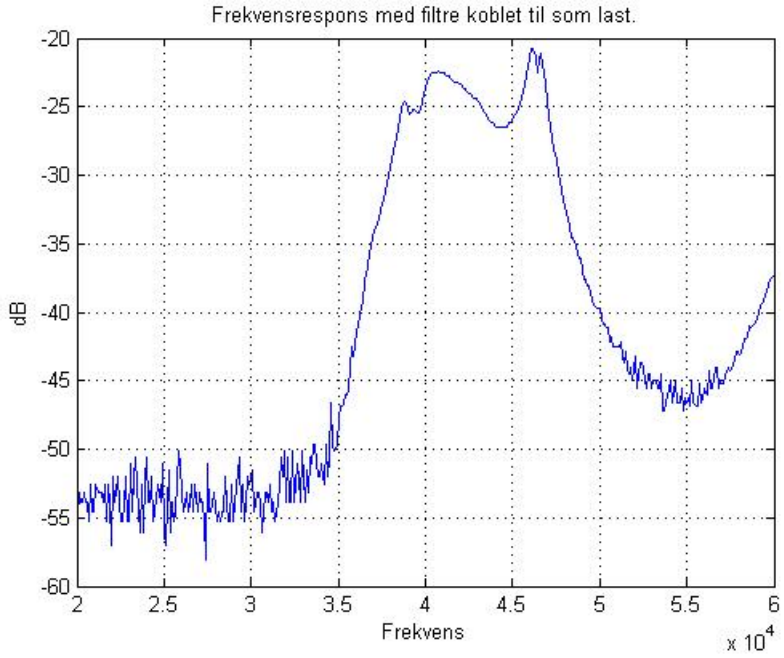
Figur 5.20: Direktivitet for ultralydsenderen. [32]



Figur 5.21: Direktivitet for ultralydmottakeren. [31]



Figur 5.22: Frekvensrespons mottaker og sender



Figur 5.23: Frekvensrespons med last tilkoblet mottakertransduseren.

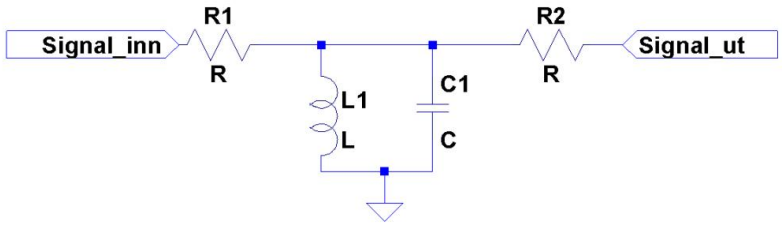
5.7 Mottakerkortet

Mottakerkortet vil bestå av to parallelle kretser som er konstruert på lik måte, som vist i figur 5.2. Forskjellen på de to kretsene ligger i det første leddet som er et båndpassfilter. Mottakertrinnet vil fungere på følgende måte: Innsignalet vil passere et av de to parallelle båndpassfiltrene, forsterkes av en operasjonsforsterker, likerettes over en diode og lagres på en kondensator. Når spenningen på en av de to kondensatorene blir høy nok vil Tmoten detektere en logisk 0 eller en logisk 1, avhengig av hvilket båndpassfilter som har sluppet igjennom signalet.

5.7.1 Valg av filtertype

Signalet som detekteres av transduseren blir sendt til to båndpassfiltre der det ene filteret har en senterfrekvens på 39,8kHz og det andre filteret har en senterfrekvens på 41,8kHz begge 1 kHz ifra senterfrekvensen på 40,8kHz, der de to frekvensene representerer logisk 1 og logisk 0. Det er flere forskjellige muligheter når det kommer til konstruksjon av filtrene. Man kan bruke passive filtre, eller aktive filtre, og det er fordeler og ulemper med de to typene.

5.7.2 Parallell LC



Figur 5.24: Kretsskjema av parallelt LC-filter

Et passivt båndpassfilter består av en seriemotstand, og en spole og en kondensator i parallell til jord som vist i figur 5.24. Bruk av

spole i filterkostruksjon gjør det mulig å lage et filter med veldig skarp knekkfrekvens i forhold til tradisjonelle RC-filtre. Filteret kalles ofte en LC-tank [10]. Navnet kommer fra det faktum at den parallelle LC-koblingen lagrer energi i spolens magnetiske felt og i kondensatorens elektriske felt. Den lagrede energien vil gå fram og tilbake mellom kondensatoren og spolen. Denne typen filter brukes typisk i frekvensområdet rundt hørbar lyd og radiofrekvenser, oftest ved radiofrekvenser. Knekkfrekvensen til et LC-filter med spole og kondensator i parallell til jord som vist på figuren beregnes ut ifra formel 5.21.

$$f_0 = \frac{1}{2\pi\sqrt{LC}} \tag{5.21}$$

Rent teoretisk vil det da måtte tas hensyn til spolens Q-verdi, dermed får man uttrykk 5.22

$$f_0 = \frac{1}{2\pi\sqrt{LC}}\sqrt{\frac{Q^2}{Q^2+1}} \tag{5.22}$$

Ved Q-verdier høyere enn 10 vil delen av uttrykket som tar hensyn til Q-verdien gi $0.995 \simeq 1$ og derfor bli overflødig i beregningene. Spolene som blir brukt i dette prosjektet har typisk en Q-verdi på over 100. Formel 5.21 er derfor et tilstrekkelig uttrykk.

Siden spolen og kondensatoren har motsatt effekt opptrer kretsen som en spenningsdeler i kombinasjon med inngangsmotstanden. Impedansen til den parallelle kretsen som går mot jord, bestående av spolen og kondensatoren, går mot uendelig ved resonansfrekvensen gitt av formel 5.21. Dermed vil hele signalet slippe udempet forbi filteret. **Ved frekvenser under resonnansfrekvensen** vil reaktansen i spolen bli lav, og reaktansen i kondensatoren bli høy. Den totale induktansen i LC-tanken vil derfor bli lik spolens induktans. **Ved frekvenser over resonnansfrekvensen** vil reaktansen i spolen bli høyere, mens reaktansen i kondensatoren blir lav. Dermed blir den totale induktansen i LC-tanken lik kondensatorens induktans.

I praksis vil det være noe tap i spolen og i kondensatoren som begrenser filterets skarphet, og som gir demping av signalet også i resonnanstoppen. Ved ønske om et slakkere LC filter kan filterets gode Q-faktor begrenses ved å sette inn en motstand i serie med spolen og kondensatoren til jord. Denne motstanden begrenser da spolens Q-faktor som igjen vil begrense filterets Q-faktor.

Det vil her bli beregnet hvilke størrelser på spolene i de to parallelle LC-tankene på mottakerkortet. Velger verdi på kondensatoren til å være

100nF siden det vil gi praktisk størrelse på spolene. Gjør så om formel 5.21 til formel 5.23 og beregner størrelsen på spolen for 39.8kHz senterfrekvens i uttrykk 5.24:

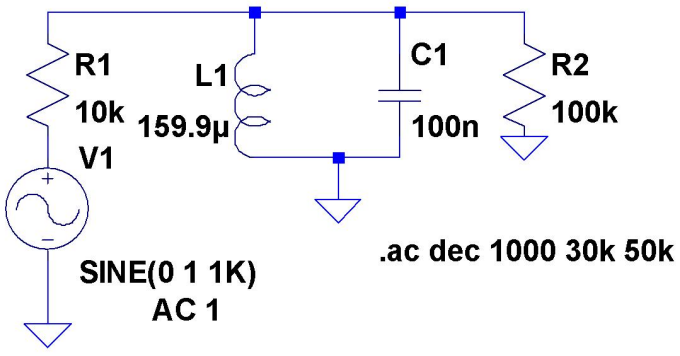
$$C_{f_o} = (\frac{1}{f_0 * 2\pi\sqrt{L}})^2 \tag{5.23}$$

$$L_{39.8kHz} = (\frac{1}{f_0 * 2\pi\sqrt{C}})^2 = (\frac{1}{39.8k * 2\pi\sqrt{100n}})^2 = 159,9uH \tag{5.24}$$

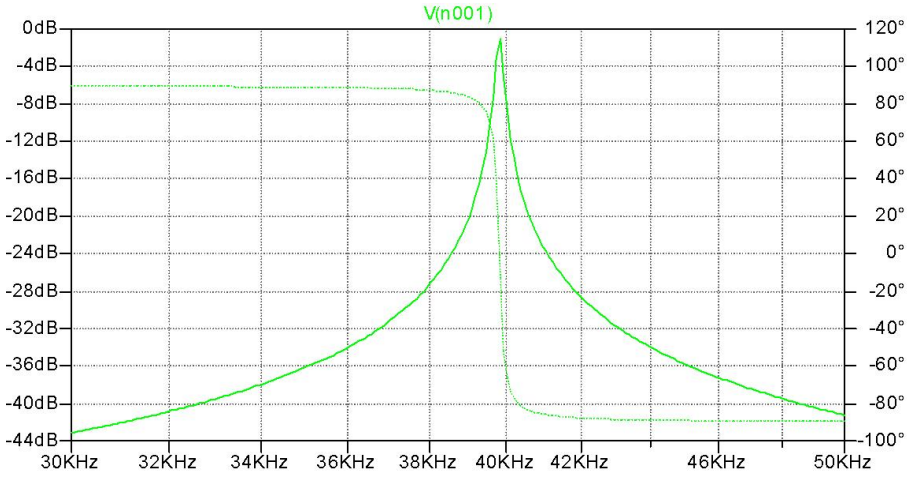
Beregner størrelsen på kondesatoren for 41.8kHz senterfrekvens i uttrykk 5.25:

$$L_{41.8kHz} = (\frac{1}{f_0 * 2\pi\sqrt{C}})^2 = (\frac{1}{41.8k * 2\pi\sqrt{100n}})^2 = 144,9uH \tag{5.25}$$

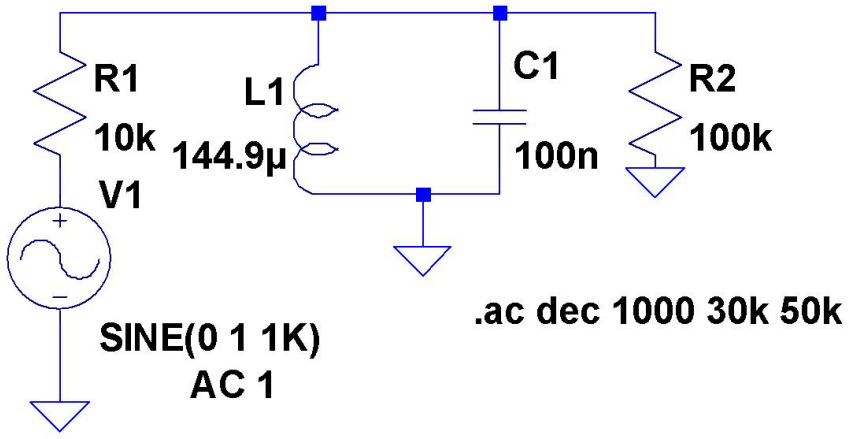
For å få et smalere passbånd på filtrene kan det brukes en større motstand i serie på filterets inngang, men signalet vil da bli svakere. Derfor blir det en vurderingssak hva som er viktigst; skarpt filter eller lav demping av signalets senterfrekvens. Til gjengjeld kan en større utgangsmotstand brukes for å kompensere for dette. Forholdet mellom inngangs- og utgangsmotstanden vil da fungere som en spenningsdeler der en større del av innsignalet vil legge seg over utgangsmotstanden ved resonnans. Simulering av filteret med de to senterfrekvensene er vist i figur 5.25 og i figur 5.27. Frekvensresponsen og faseresponsen til de to filtrene er vist i figur 5.26 respektive i figur 5.28.



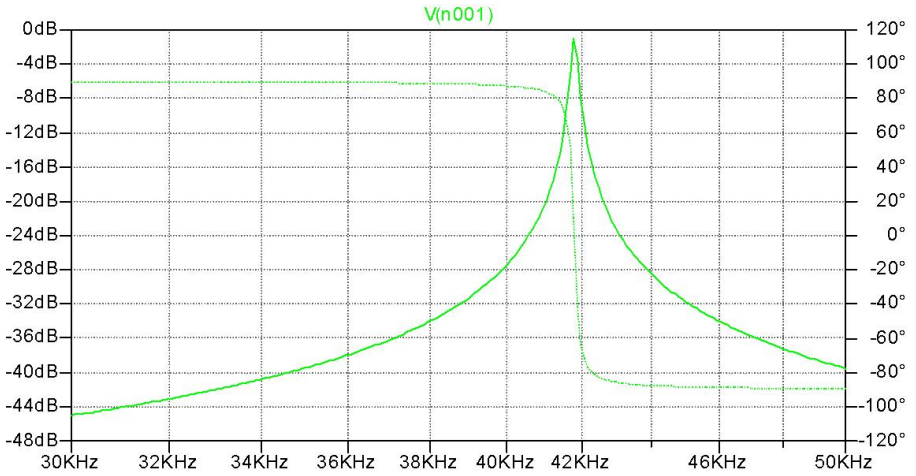
Figur 5.25: Kretsskjema for 39,8kHz parallell-LC filter



Figur 5.26: Frekvens- og faserespons for 39,8kHz parallell LC-filter

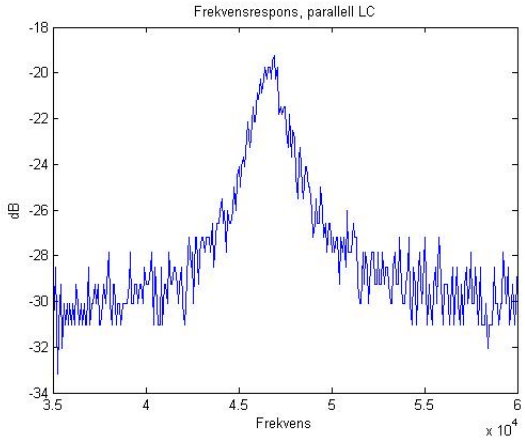


Figur 5.27: Kretsskjema for 41,8kHz parallell-LC filter.



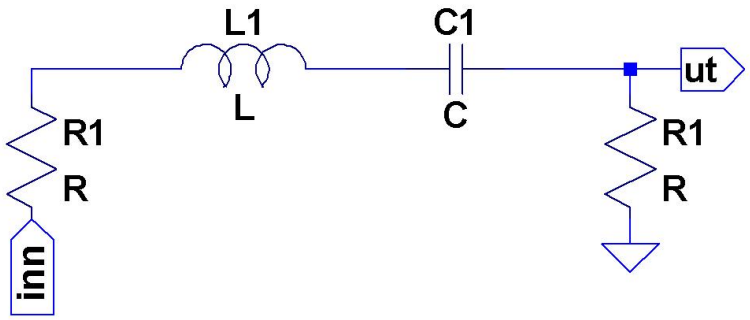
Figur 5.28: Frekvens- og faserespons for 41,8kHz parallell-LC filter.

Det parallelle LC-filteret med passende komponentstørrelser ble utprøvd og målt med tilgjengelige måleinstrumenter. Filterets målte frekvensrespons er vist i figur 5.29. Da det ikke var tilgjengelig spoler på 159,9μH eller 144,9μH ble det benyttet en 141μH spole under testingen. Da spolens Q-verdi var den samme, og forskjellen i størrelse fra beregnete verdier var ca 4μH er dette en ikke-signifikant forskjell, og båndbredde og demping må kunne sies å være representativ. -3dB båndbredden leses av figur 5.29 til å være om lag 1,9kHz. Man ser også at filterets senterfrekvens dempes med om lag 19dB, noe som må sies å være en betydelig demping. Dette gjør også at innsignalet blir vanskeligere å lese nøyaktig med oscilloskopet, dermed blir grafen ujevn. Dette signalet vil kreve en høy forsterkning for å passe til likerretter- og detekteringsapplikasjonen i dette prosjektet. Med høy forsterkning av et svakt signal betyr det at også støy i kretsen vil forsterkes opp og kunne påvirke signalene, med økt feilrate i dataoverføringen som resultat.



Figur 5.29: Målt frekvensrespons for parallell LC-filter.

5.7.3 Seriell LC



Figur 5.30: Kretsskjema for serielt LC-filter.

Et serielt LC filter som vist i figur 5.30 utnytter resonnansegenskapene til en LC-konfigurasjon på samme måte som et parallell LC-filter. En spole og en kondensator koblet i serie vil resonnere ved en bestemt frekvens gitt av komponentenes størrelse, og gi en impedansefri signalvei ved denne frekvensen. Når inngangsfrekvensen er lik resonnansfrekvensen vil signalet slippes udempet igjennom seriekoblingen. Frekvenser som ligger under resonnansfrekvensen vil dempes av kondensatorens motstand mot lavfrekvente signaler, og frekvenser som ligger over resonnansfrekvensen

vil dempes av spolens motstand mot høyfrekvente signaler. I forhold til et parallell LC-filter vil uttrykket for båndbredden av et serielt LC-filter være det inverse. Det vil være nyttig når det skal avgjøres om seriell eller parallell filterkonfigurasjon er mest hensiktsmessig. Ut ifra inngangs- og utgangsmotstand, spolens Q-faktor og kravet til filterets Q-faktor [8] kan man avgjøre hvor stor spole det er behov for. Dette beregnes med formel 5.26:

$$Q_{bp} = \frac{f_0}{BW_{3dB}} \tag{5.26}$$

f_0 er senterfrekvensen, og BW_{3dB} er båndbredden mellom 3 dB demping av signalet i forhold til senterfrekvensen.

Videre beregnes størrelsen på spolen i kretsen ut ifra tilgjengelig Q-verdi på spolen Q_L, ønsket Q-faktor Q_{bp}, ut ifra inngangsmotstand R_s og utgangsmotstand R_L med formel 5.27.

$$L = \frac{R_S + R_L}{\omega_0 \frac{1}{Q_b p} - \frac{1}{Q_L}} \tag{5.27}$$

Derfra beregnes kondensatorens størrelse med den kjente formelen for serielle LC-filtres udempede resonansfrekvens, på samme måte som for parallelle LC-filtre, med formel 5.28.

$$f_0 = \frac{1}{2\pi\sqrt{LC}} \tag{5.28}$$

Den serielle LC-koblingen fungerer som en resonator, og ved resonans er koblingens totale, komplekse impedans uttrykt med formel 5.29.

$$Z_{LC} = Z_L + Z_C = 0 \tag{5.29}$$

der spolens impedans er uttrykt med formel 5.30 [21] og kondensatorens impedanse er uttrykt med formel 5.31

$$Z_L = L * s \tag{5.30}$$

$$Z_C = \frac{1}{Cs} \tag{5.31}$$

Den dempede resonansfrekvensen uttrykkes ut ifra resonatorens naturlige frekvens og dempingsfaktoren i formel 5.32.

$$\omega_d = \sqrt{(\omega_0)^2 - (\zeta)^2} \tag{5.32}$$

I og med at alle de tre komponentene inngangsmotstand, spole og kondensator er i serie med kilden i en seriell LC-krets kan dempingsfaktoren uttrykkes med formel 5.33

$$\zeta = \frac{R\sqrt{C}}{2\sqrt{L}} \tag{5.33}$$

Beregning av komponentstørrelser blir gjort både for en senterfrekvens på 39.8kHz og på 41.8kHz, med tilsvarende krav for begge filtrene. For å sikre at svingerens senterfrekvens på 40.8kHz ikke har større påvirkning på komparatorkretsene enn filterets senterfrekvens blir det satt som krav at filteret skal ha en 3dB båndbredde på 1000Hz, dvs at signalet skal dempes med maksimalt 3dB 500Hz over og under senterfrekvensen.

For 39.8kHz: Beregner i uttrykk 5.34 filteres Q-verdi med formel 5.26.

$$Q_{bp} = \frac{39.8kHz}{BW_{1kHz}} = 39.8 \tag{5.34}$$

Beregner i uttrykk 5.35 videre hvor stor spole det er behov for med formel 5.27. Fra datablad for spoler fra no.farnell.com er det konstatert at tilgjengelig Q-verdi på spolen = 100, inngangs- og utgangsmotstand er valgt til å være 100 Ω:

$$L = \frac{100 + 100}{(2\pi 39.8kHz)(\frac{1}{39.8} - \frac{1}{100})} = 52.88mH \tag{5.35}$$

Nærmeste standard størrelse spole er på 56mH, så derfor velges denne ved videre beregninger.
Beregner i uttrykk 5.36 størrelsen på kondensator med formel 5.28.

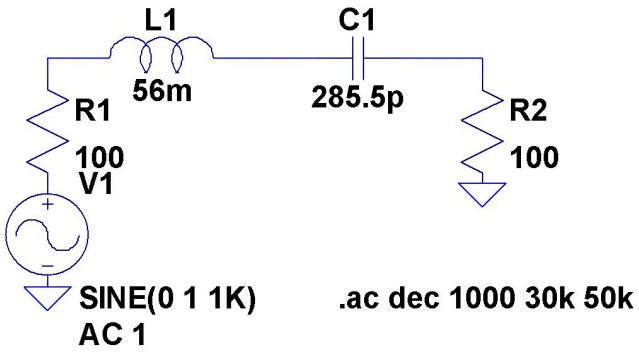
$$C_{39.8kHz} = (\frac{1}{2\pi f_0\sqrt{L}})^2 = (\frac{1}{39.8k * 2\pi\sqrt{56m}})^2 = 285.55pF \tag{5.36}$$

For 41.8kHz gjøres beregningene på tilsvarende måte.:
Beregner i uttrykk 5.37 filteres Q-verdi med formel 5.26.

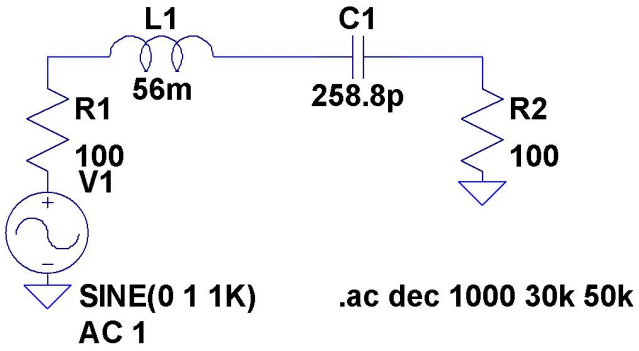
$$Q_{bp} = \frac{41.8kHz}{BW_{1kHz}} = 41.8 \tag{5.37}$$

Beregner i uttrykk 5.38 videre hvor stor spole det er behov for med fomel 5.27. Også her er tilgjengelig Q-verdi på spolen = 100. Inngangs- og utgangsmotstand velges også her til 100 Ω:

$$L = \frac{100 + 100}{(2\pi 41.8kHz)(\frac{1}{41.8} - \frac{1}{100})} = 54.69mH \tag{5.38}$$



Figur 5.31: Kretsskjema for 39,8kHz serie LC-filter



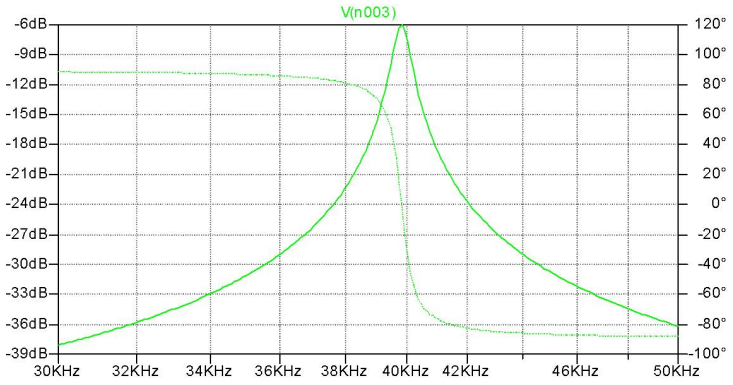
Figur 5.32: Kretsskjema for 41,8kHz båndpassfilter

Nærmeste standard størrelse spole er på 56mH, så derfor velges denne ved videre beregninger.
Beregner i uttrykk 5.39 størrelsen på kondensator med formel 5.28:

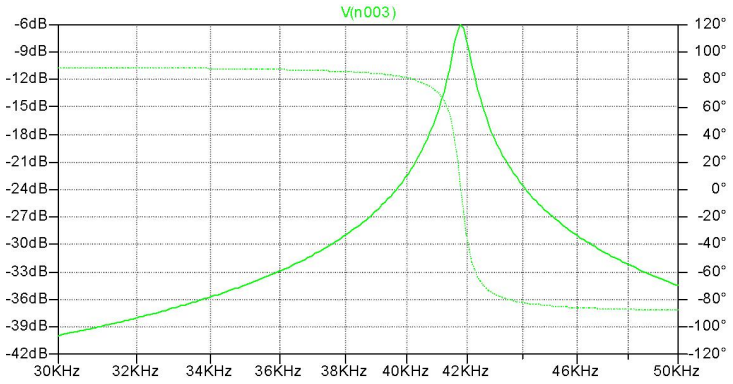
$$C_{41.8kHz} = (\frac{1}{2\pi f_0 \sqrt{L}})^2 = (\frac{1}{41.8k * 2\pi\sqrt{56m}})^2 = 258.88pF \qquad (5.39)$$

Simulering av filteret med de to senterfrekvensene er vist i figur 5.31 og i figur 5.32. Frekvensresponsen til de to filtrene er vist i figur 5.33 respektive i figur 5.34.

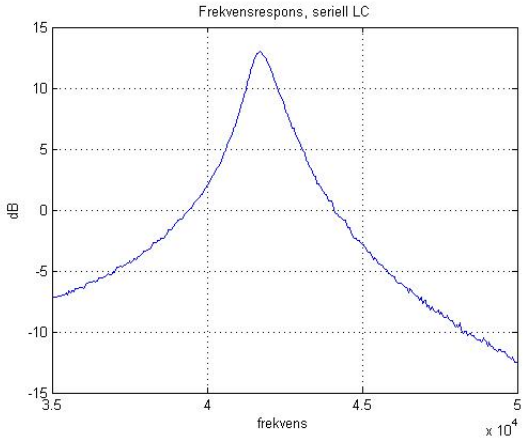
Figur 5.33 viser en simulert -3dB båndbredde på 42,09kHz - 41,52kHz = 0,57kHz for 39,8kHz-filteret. Figur 5.34 viser en simulert -3dB båndbredde på 40,08kHz - 39,53kHz = 0,55kHz, noe som er bedre enn kravet om en -



Figur 5.33: Simulert frekvensrespons for 39,8kHz LC serie-filter



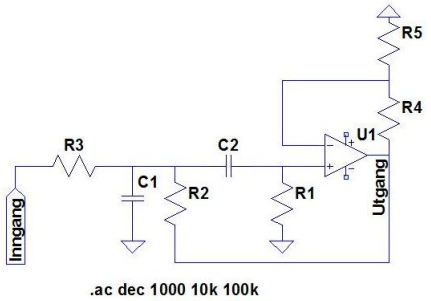
Figur 5.34: simulert frekvensrespons for 41,8kHz LC serie-filter



Figur 5.35: Målt frekvensrespons for serielt LC-filter

3dB båndbredde på 1000Hz. Denne feilen kan tilskrives at simuleringen ikke tar hensyn til Q-verdien til komponentene, men behandler alle komponenter som ideelle. I utregningene for komponentstørrelser er det derimot tatt høyde for komponentenes Q-verdi som vil begrense filtrenes skarphet.

I figur 5.35 vises den målte frekvensresponsen til det serielle LC-filteret. Filteret i testen har en senterfrekvens på 41,8 kHz, og fra figuren leses det av en -3dB båndbredde på ganske nøyaktig 1 kHz. Filteret oppfører seg med andre ord helt etter spesifikasjonene. Figuren viser også at filteret faktisk forsterker inngangssignalet. Denne forsterkningen kan tilskrives de høye spenningene som genereres i spolen ved resonnansfrekvensen.



Figur 5.36: Kretsskjema for Sallen-Key båndpassfilter

5.7.4 Sallen-Key

Et alternativ til passive filtre er å bruke et båndpassfilter som ikke har noen spoler, som et Sallen-Key båndpassfilter [1, 3] med aktive komponenter, vist i figur 5.36. Den aktive komponenten er her en operasjonsforsterker. Ved å bruke en operasjonsforsterker som en del av filterkonstruksjonen kan spolene i filteret fjernes, samtidig som et skarpt filter fortsatt oppnås. Det kan altså lages et skarpt filter der ikke bør tas hensyn til spoler, som er unøyaktige og ulineære. Aktive filtre kan brukes til å lage alle slags filtre: Lavpass, høypass, båndpass og båndstopp. Filteret kan også tilpasses andre behov, som flatt passbånd, uniform tidsforsinkelse og skarphet mellom passbånd og stoppbånd. I dette prosjektet er behovet for flatt passbånd og uniform tidsforsinkelse eller fase fraværende, da hvert filter ideelt bare skal slippe igjennom en frekvens og fullstendig dempe alle andre frekvenser. Høy Q-faktor er derfor det viktigste i dette prosjektet. Det finnes mange alternative konfigurasjoner for Sallen-Key båndpassfiltre, der hver konfigurasjon vil ha sine fordeler, som f.eks flatt passbånd eller flatt stoppbånd, eller høy Q-faktor. I dette prosjektet er det lagt vekt på å lage filteret så skarpt som mulig, altså å få en høy Q-faktor.

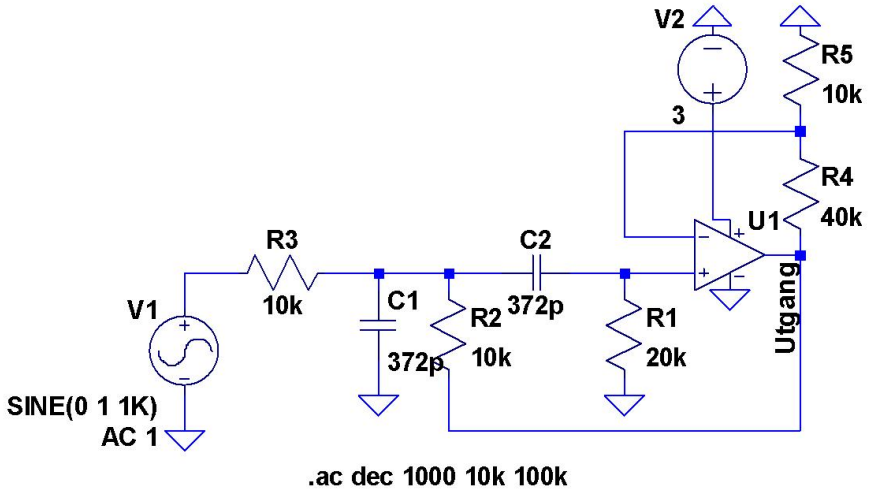
Filterets senterfrekvens beregnes med formel 5.40:

$$\omega_0 = \frac{1}{\sqrt{R_1 R_2 C_1 C_2}} \tag{5.40}$$

Filterets Q-verdi beregnes med formel 5.41:

$$Q = \sqrt{\frac{R_2}{R_1}} \frac{\sqrt{C_1 C_2}}{C_1 + C_2} \tag{5.41}$$

Båndbredden uttrykkes med formel 5.42.



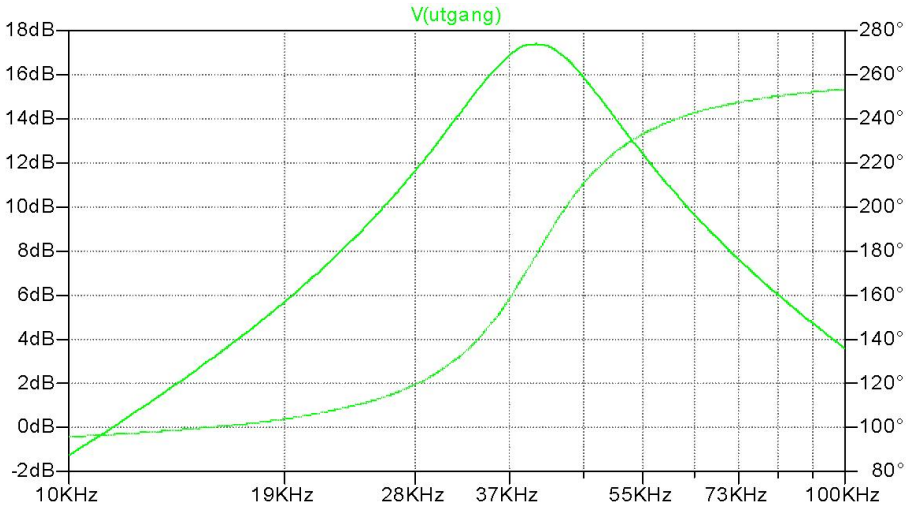
Figur 5.37: Oppkobling av et 39kHz Sallen-Key båndpassfilter

$$BW = \frac{2}{R_2 * C}, C_1 = C_2 \tag{5.42}$$

Ved frekvenser mye høyere enn og mye lavere enn ω_0 vil filterresponsen oppføre seg som et førsteordens høypass- eller lavpassfilter og demper da signalet med 20dB pr. dekade [1]. Størrelsen på motstandene velges typisk til å ligge i kΩ-området for å begrense strømtrekket.

Størrelsen på alle komponenter i oppkoblingen beregnes for å nå ønsket senterfrekvens. Det er her valgt å kjøre en filtersimulering med 39kHz som filterets senterfrekvens. Kretsskjema er vist i figur 5.37, og simulert frekvensrespons er vist i figur 5.38. Da *Analog Devices* har en utmerket kalkulator tilgjengelig for beregning av komponentstørrelser i et Sallen-Key båndpassfilter, har denne blitt brukt som veiledning ved beregning av komponentstørrelser.

Figur 5.38 viser at et aktivt Sallen-Key båndpassfilter vil ha et mye bredere passbånd enn et passivt filter med spoler. Faktisk vil filteret ha et -3dB passbånd på over 8kHz, og kanalseparasjonen mellom de to sendefrekvensene på 39kHz og 41kHz er i praksis fraværende. Dermed er konklusjonen at et sallen-key båndpassfilter er mindre egnet i prosektet enn et passivt LC båndpassfilter.



Figur 5.38: Simulert frekvenssespons for et 39kHz sallen-key båndpassfilter

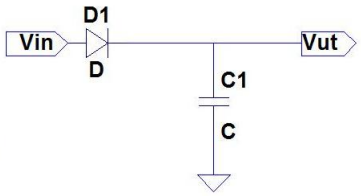
5.7.5 Andre alternativer

Et siste alternativ kan være å bruke et lavpassfilter for signalet med lavest frekvens, og et høypassfilter for signalet med høyest frekvens. Ulempen med dette er at da vil alle signaler under/over knekkfrekvensen slippes igjennom og trigge mottak av bit i utide. Dette kan likevel fungere, da svingeren som brukes er laget med en resonnansfrekvens på 40kHz. Man kan se av senderens og mottakerens frekvenskarakteristikk at signalet raskt dempes utenfor resonnansfrekvensen. Fordelen med passive høypass- og lavpassfiltre er at det kan bygges passive filtre som ikke trekker energi fra Tmotens begrensede strømforsyning og man slipper å bruke ulineære og upresise spoler i filteret. Skal et høypass- eller et lavpassfilter lages skarpt nok for denne oppkoblingen må det i midlertid brukes aktive komponenter i filteroppkoblingen.

5.7.6 Forsterker

Hvis signalene fra ultralydmottakeren blir veldig lave, vil det være behov for en forsterker på inngangen. Forsterkeren bør plasseres mellom filteret og likeretteren [2]. En viktige egenskap for forsterkeren vil være at den fungerer godt i hele området mellom forsterkerens forsyningsspenninger. Det ble først gjort forsøk uten forsterker, da denne trekker effekt, og begrenser batteriets levetid, men det viste seg fort at behovet for en forsterker absolutt var til stede.

Selv om en operasjonsforsterker i prinsippet skal fungere optimalt i hele området mellom forsyningsspenningene, vil det alltid være begrensninger i nærheten av forsyningsspenningene. I dette prosjektet finnes 3V og Gnd tilgjengelig. Signaljord bør derfor legges til $\frac{Vcc}{2}$, 1,5 V. Siden det ikke finnes en spenning på 1,5 V, tilgjengelig lages dette med to motstander koblet som en spenningsdeler der det er nødvendig. For å få en 100 Ω motstand på filterets utgang legges en 200 Ω motstand til Vcc og en 200 Ω motstand til Gnd. Da Vcc og Gnd signalmessig er likeverdige kan de to motstandene regnes som en parallellkobling. Beregner dermed utgangsmotstanden til filteret til å være $\frac{R_{Gnd}*R_{Vcc}}{R_{Gnd}+R_{Vcc}} = \frac{200*200}{200+200} = 100\Omega$. Dette vil gi signaljord som følger eventuelle variasjoner i forsyningsspenningen. Signaljorden vil altså alltid ligge på Vcc/2. Samtidig vil dette gi en "vei" på 400 Ω mellom Vcc og Gnd, noe som vil gi et ekstra strømforbruk på $\frac{3V}{400\Omega} = 7,5mA$ pr. filter. For å begrense strømtrekket kan høyere inngangs- og utgangsmotstand på filteret brukes. På samme måte må tilbakekoblingen på forsterkerens utgang legges til signaljord. Også her brukes en spenningsdeler mellom Vcc og Gnd. Denne motstanden må være liten i forhold til forsterkerens tilbakekobling. Hvis motstanden nærmer seg motstanden i tilbakekoblingen vil ikke spenningsdeleren se to like store motstander, og dermed vil ikke signaljord ligge på samme nivå som på forsterkerens inngang. Motstander på 1/10 av tilbakekoblingens minste motstand, eller 1/100 av den totale motstanden i tilbakekoblingen er derfor hensiktsmessig. De to motstandene er dermed på 3kΩ hver, noe som gir et strømtrekk på $\frac{3V}{6k\Omega} = 500\mu A$



Figur 5.39: Diodelikeretter med glattekondensator

5.7.7 Likeretterdelen

Det filtrerte signalet fra svingeren forsterkes opp og likerettes for å lade opp en kondensator i *integratoren* omtalt i avsnitt 5.7.8. Når det ligger en tilstrekkelig høy spenning over kondensatoren trigges en av de to inngangene i Tmote Sky sensornoden og detekterer en logisk 0 eller 1.

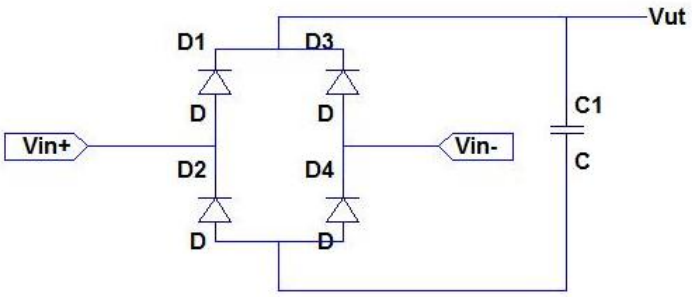
Halvperiodelikeretter

Signalet kan likerettes med en vanlig halvperiodelikeretter bestående av en enkel diode, og lagres over en kondensator som vist i figur 5.39. Dioden vil da ha den funksjonen at den slipper igjennom signalet når spenningsforskjellen mellom anode- og katodesiden ligger høyere enn diodens ledespenning, mens når spenningsforskjellen ligger lavere enn diodens ledespenning vil signalet sperres. Dette forhindrer at spenningen på kondensatoren det integreres opp spenning over trekkes ned, og signalet likerettes. Et diodedropp er på ca 0.7 V, noe som betyr at signaler med spenning lavere enn et diodedropp ikke vil slippe forbi. Den gjennomsnittlige utspenningen dersom et konstant sinussignal sendes igjennom en halvperiodelikeretter [5] kan beregnes med formel 5.43.

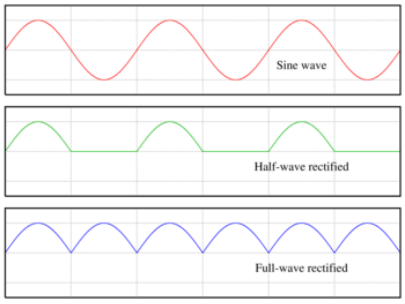
$$U_{ut} = \frac{\frac{2U_a}{\pi}}{2} - U_d = \frac{U_a}{\pi} - U_d \tag{5.43}$$

Brolikeretter

Alternativet til en halvperiodelikeretter er en brolikeretter som består av fire dioder, vist i figur 5.40. Brolikeretteren vil slippe igjennom både positive og negative signaler som positive signaler, noe som kan redusere antallet perioder av et signal som sendes pr. sendte bit for å nå en satt spenning. Virkemåten til en brolikeretter er følgende: Når innsignalet er positivt og høyere enn to diodedropp vil signalet slippes igjennom diode



Figur 5.40: Brolikeretter



Figur 5.41: Viser inngang, utgang fra halvperiodelikeretter og utgang fra brolikeretter. [40]

D_1 og D_4. Når innsignalet derimot er negativt og lavere enn -2 diodedropp vil signalet slippes forbi diode D_2 og D_3. Dermed likerettes både den positive og den negative delen av signalet, og utsignal vil bli som vist i figur 5.41. Ulempen med denne er at den har to dioder i serie, som gir to diodedropp i spenningen for positive og negative signaler. Terskelen for å slippe signalene forbi likeretteren blir dermed høyere, og et sterkere signal må til. Utspenningen til en brolikeretter kan uttrykkes med formel 5.44, der U_a er innspenningen og U_d er et diodedropp.

$$U_{ut} = \frac{2U_a}{\pi} - 2U_d \tag{5.44}$$

Ved bruk av forsterker vil det også her måtte genereres signaljord på $\frac{V_{cc}}{2}$ til brolikeretteren, utladningsmotstanden og kondensatoren, da brolikeretteren likeretter et signal som svinger om en felles referansespenning. De ekstra komponentene, og det ekstra strømtrekket det medfører å måtte

generere signaljord også her, i tillegg til det ekstra diodedroppet en brolikeretter medfører gjør at brolikeretteren blir et mindre atraktivt alternativ i forhold til å bruke en enkel halvperiodelikeretter. Et ekstra diodedropp, målt til ca 0,5 V innebærer i dette prosjektet at signalet må opp i minst 1/3 av maksimal signalstyrke før signalet kan utløse noen reaksjon hos komparatoren omtalt i avsnitt 5.7.9. Når forsterkeren har en maksimal amplitude på 1,5 V vil en brolikeretter sperre signalet helt til forsterkeren når 2/3 av maksimal amplitude. Derfor er det i dette prosjektet brukt en enkel halvperiodelikeretter. Likeretteren er også testet, og fungerer tilfredsstillende.

Ved å bruke en glattekondensator på katodesiden av de to likeretterkretsene kan man få en jevnere likespenning som kan brukes til å drive en krets, eller som i dette prosjektet, i en komparator. Utspenninga til en enkel halvperiodelikeretter kan ved bruk av kondensatoren på utgangen uttrykkes med formel 5.45:

$$U_{ut} = U_a - U_d \tag{5.45}$$

5.7.8 Integratoren

Det likerettede signalet til mottakeren lagres som nevnt på en kondensator for å utløse en komparator som igjen utløser et avbrudd i Tmote Sky sensornoden som da detekterer et mottatt bit. Det lagrede signalet skal deretter lades ut, og deretter lades opp igjen når et nytt signal mottas. Utladningen skjer enkelt via en motstand som er koblet til jord. Utladningsmotstanden (R) er avhengig av utladningstiden (t), størrelsen på lagringskondensatoren (C), spenningen før utladning (V_i), spenningen ved fullstendig utladning (V_F), og ønsket spenningsfall (v), og er uttrykt med formel 5.47 utledet [22] fra formel 5.46.

$$v = V_F + (V_i - V_F)e^{\frac{-t}{\tau}} = V_F + (V_i - V_F)e^{\frac{-t}{RC}} \tag{5.46}$$

$$R = -\frac{t}{\ln(\frac{v-V_F}{V_i-V_F}) * C} \tag{5.47}$$

I Tmote Sky sensornodene blir det utløst et avbrudd som følge av når en av de to komparatorene omtalt i 5.7.9 sender ut en positiv flanke. Når et avbrudd er utløst vil en asynkron hendelse i Tmote Sky sensornoden starte. Dermed omgås problemet med tidsoppløsningen på 1 ms nøyaktighet. Likevel bør utladningsmotstanden være stor, da forsterkerkretsen kontinuerlig holder utgangen på 1,5 V når det ikke

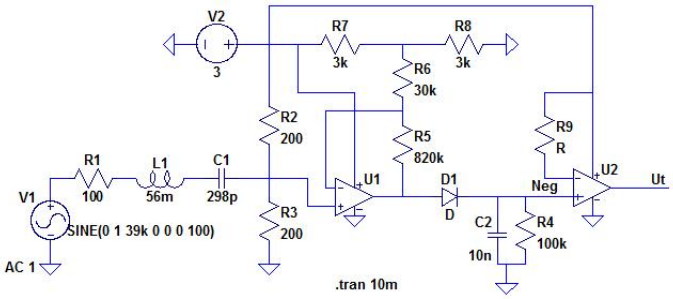
mottas noe signal. En spenning på 1,5 V - et diodedropp vil derfor hele tiden føre til at det går en strøm igjennom utladningsmotstanden. Denne motstanden bør, for å begrense strømtrekket, være størst mulig. Samtidig vil høy utladningsmotstand gjøre det lettere å lade opp kondensatoren når et signal blir mottatt. Basert på at TMote Sky ikke kan gi bedre tidsoppløsning enn 1 ms, er det konkludert med at hvert utsendte bit må sendes i minst 1 ms. Kretsen har deretter en oppladningstid på 3 ganger sendetiden, altså 3 ms før et nytt signal kan sendes. Ved å legge til ytterligere 1 ms som margin på oppladningstiden, gir dette at et signal kan sendes hver femte ms, med 5 ms - 1 ms = 4 ms til rådighet for utladning av integratoren fra maksimal spenning til tilnærmet hvilespenning, altså fra 3 V - 0,5 V = 2,5 V til tilnærmet 1,5 V - 0,5 V. Et diodedropp er målt til 0,5 V. 90% utladning av signalet, til 1,15 V i løpet av 3,5 ms av de 4 ms tilgjengelig er vurdert til å være tilstrekkelig utladning for å kunne detektere et nytt avbrudd. Størrelsen på utladningsmotstanden er derfor beregnet med 3,5 ms utladninstid med formel 5.47 i formel 5.48.

$$R = -\frac{3,5m}{\ln(\frac{1,15V-1,0V}{2,5V-1,0V}) * 10nF} = 152k\Omega \tag{5.48}$$

Motstanden på 152kΩ gir et ekstra strømtrekk på $\frac{1}{152k\Omega} = 6,6\mu A$, fra forsterkeren som kontinuerlig gir ut 1,5 V, 1,0 V på katodesiden av dioden. Dette strømtrekket må kunne sies å være ubetydelig, da kretsen kontinuerlig trekker 14 mA (målt). For å senke strømtrekket via motstanden ytterligere kan eventuelt kondensatoren gjøres mindre, og motstanden tilsvarende større. Ved å bruke en 1 nF kondensator vil utladningstida bli den samme med en 1,52MΩ motstand. Strømtrekket via motstanden kan på den måten begrenses til 1/10 av det beregnede, altså til 0,66 μA. For å beholde en viss nøyaktighet i RC-leddet blir likevel motstanden begrenset til 152kΩ, og kondensatoren forblir 10 nF.

5.7.9 Komparatoren

Da det ble valgt at *HplMsp430InterruptC*-grensesnittet i TinyOS 2.0 skulle brukes, kom behovet for en analog komparator inn i bildet. Denne gir ut en positiv flanke rask nok til at den blir detektert som en flanke i Tmote Sky sensornoden. Komparatorkjernen [29] er en enkel ST Microelectronics TS912 [23] operasjonsforsterker. Komparatorens positive inngang er koblet til integratorens utgang, den negative inngangen er koblet til V_{cc} via et potensiometer for justering av spenningsnivå, og utgangen er koblet til Tmote Sky sensornodens inngang som vist i

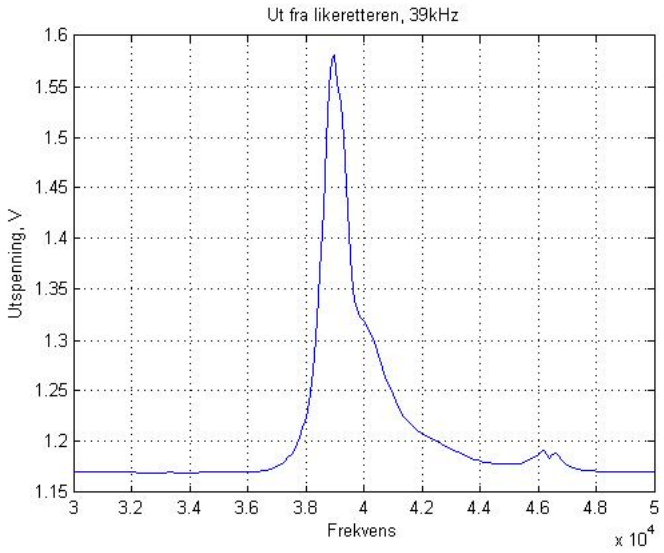


Figur 5.42: Kretsskjema for den ene av de to inngangskanalene på mottakerkortet.

figur 5.42. Ved å justere potensiometeret på den negative inngangen vil spenningen på den inngangen reguleres. Da komparatoren ikke har noen tilbakekobling vil den hele tiden gi ut 0 V eller 3 V og svitsje kun mellom disse to verdiene. Det betyr at når integratoren, som er koblet til den positive inngangen av komparatoren, når en høyere spenning enn den justerte innspenningen på negativ inngang, vil utgangen svitsje fra 0 V til 3 V og trigge et avbrudd på en av de to inngangene på Tmoten. I det spenningen ut fra integratoren faller under spenningen på den negative inngangen vil kanalen være klar for å motta et nytt bit. Kretsskjema for en komplett inngang med alle komponenter er vist i figur 5.42. Hvert kort har 2 slike innganger koblet parallelt.

5.7.10 Test og justering av mottakerkortet

Mottakerkortet ble utprøvd med hensyn til strømtrekk og kanalseparasjon på inngangen ved å plassere en ultralydsender 1.0 cm ifra transduseren på mottakerkortet og steppe et 10 Volts peak-to-peak signal fra 30kHz til 50kHz. Først ble det testet med en forsterkning på 9,3 i forsterkerkretsen på hver av inngangene, og utspenningen fra hver kanal. I figur 5.43 vises signalet til Tmote Skys analoge inngang som skal detektere et 39kHz signal, og i figur 5.44 vises signalet til Tmote Skys analoge inngang som skal detektere 41kHz signal. De to figurene viser at filtreringen og kanalseparasjonen fungerer bra med 39kHz filteret. Forskjellen på å ikke detektere signal, og å detektere signal kunne derimot vært høyere. For å øke forskjellen mellom innspenningen ved mottatt signal vs. ikke mottatt signal ble forsterkningen justert ved å øke motstand R5 i figur 5.42. For begge inngangene ble motstanden justert gradvis opp fra 250kΩ til et nivå der kanalene ikke kunne separeres lengre. Motstanden ble da justert litt ned igjen, og ble da målt til 500kΩ. Som følge øker da forsterkningen på

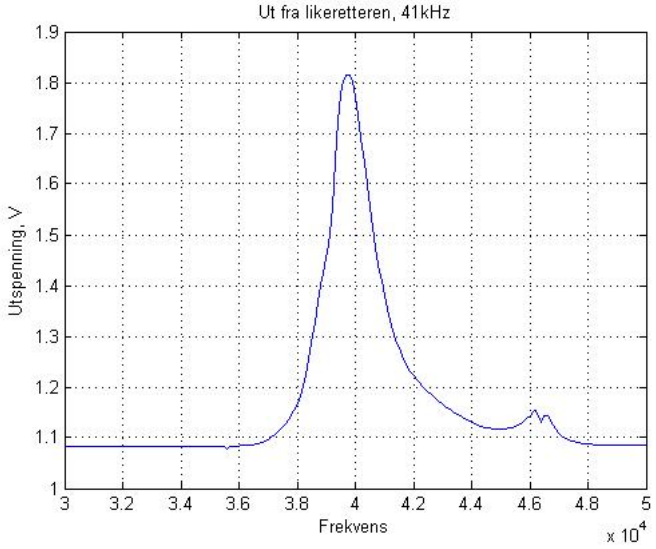


Figur 5.43: Inngangen på Tmote Sky, 39k.

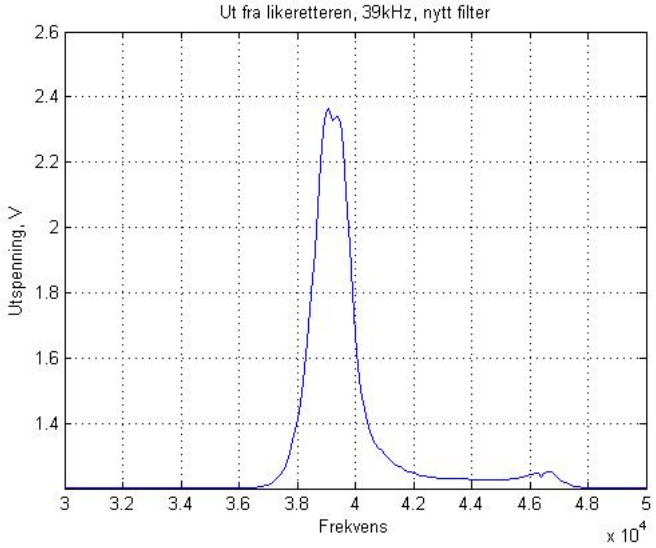
begge kanalene til 17,67.

Da kanalen for 41kHz signal ga en spenningstopp på utgangen ved ca 39.9kHz ble det utprøvd med en mindre kondensator på 220pF på seriell LC-filteret for å øke senterfrekvensen til dette filteret. Formel 5.28 gir en senterfrekvens på 45,3kHz med den mindre kondensatoren. Dette er 3,5kHz høyere frekvens enn ønsket senterfrekvens. En lik test ble deretter gjort på de to inngangene. 39kHz-inngangen ble testet på nytt for å se om endringer på filteret med høyest frekvens hadde noen påvirkning på filteret med lavest frekvens. Resultatet av testen med det nye filteret er vist i figur 5.45 for 39kHz-kanalen, og i figur 5.46 for 41kHz inngangen. Figur 5.47 viser de to kanalene fra figur 5.45 og figur 5.46 i samme plot, og det kommer tydelig fram at kanalseparasjonen er for dårlig for å skille de to frekvensene fra hverandre. Senterfrekvensen til de to mottakerfiltrene må dessuten samsvare med senterfrekvensen til de to oscillatorene hos senderkortene.

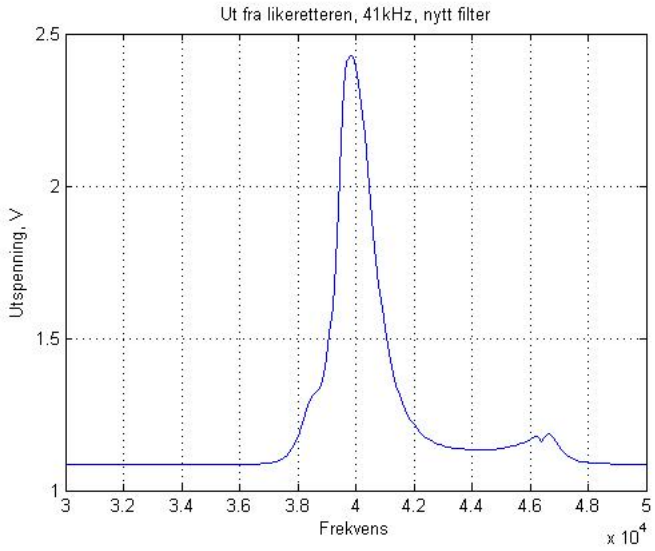
Etter testing med filtre med forskjellige senterfrekvenser ble den endelige konklusjonen at de opprinnelige filtrene, med beregnet knekkfrekvens på 39,8kHz og 41,8kHz, var de som ville fungere best. Grunnen til at de to opprinnelig beregnede knekkfrekvensene var de som fungerte best kan nok tilskrives det at de ligger like langt ifra filterets senterfrekvens. De to filtrene vil da belaste ultralydmottakeren likeverdig over og under senterfrekvensen og derfor gi den best mulige kanalseparasjonen. Dermed ble de to opprinnelige filtrene igjen satt inn. De to forsterkerne ble deretter fin-



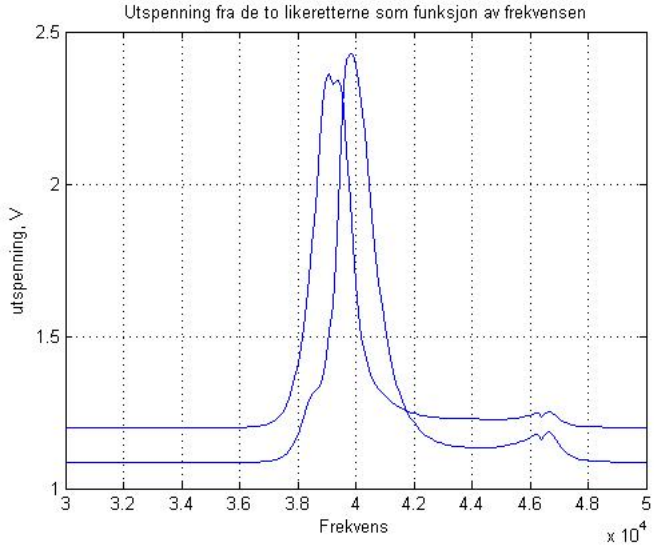
Figur 5.44: Inngangen på Tmote Sky, 41k.



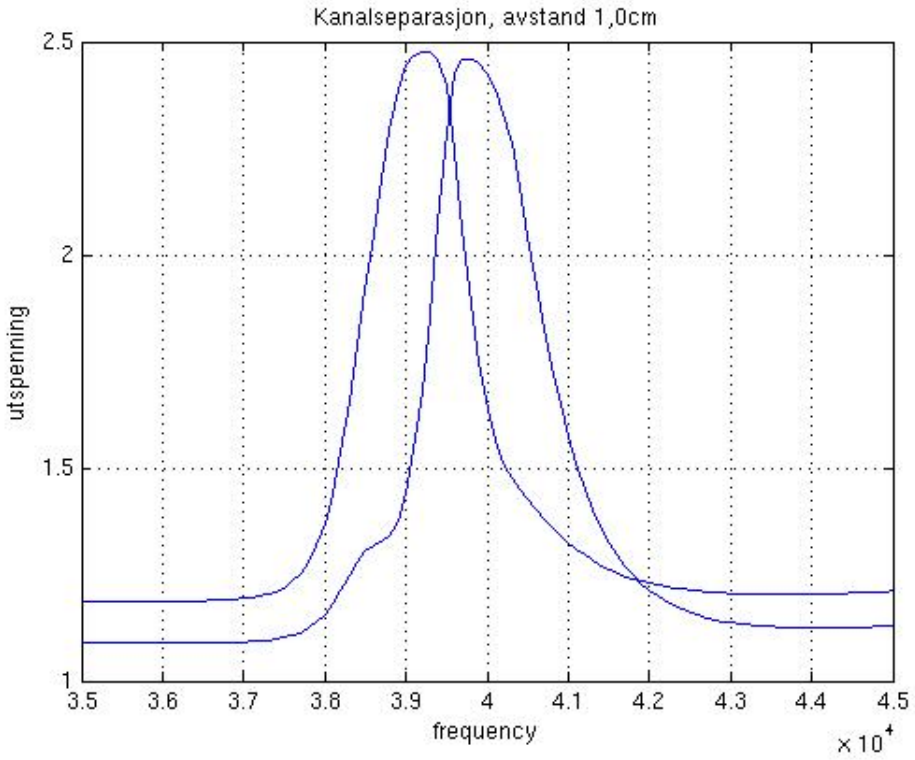
Figur 5.45: Inngangen på Tmote Sky, 39kHz



Figur 5.46: Inngangen på Tmote Sky, 41kHz



Figur 5.47: De to inngangene på Tmote Sky.



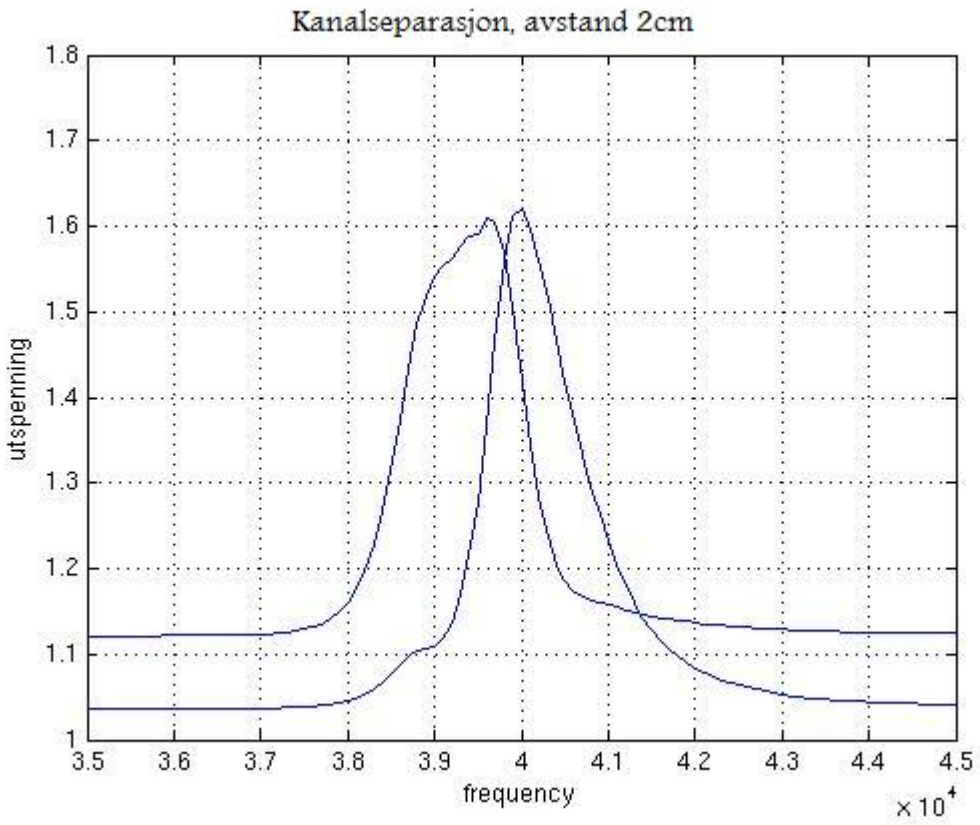
Figur 5.48: Kanalseparasjon ved 1 cm avstand mellom transduserne.

justert til de to kanalene nådde et like høyt signal samtidig som kanalseparasjonen var så god som mulig. Ved 1 cm avstand mellom sender- og mottakertransduseren ble kanalseparasjonen, etter finjustering av forsterkerne, som vist i figur 5.48. Potensiometrene i forsterkerens tilbakekobling, R5 i figur 5.42, ble deretter målt til å være 500kΩ for begge de to inngangene.

Av figur 5.48 blir det lest av ved hvilke frekvenser det er størst mulig kanalseparasjon. Deretter ble det ut ifra dette valgt to nye senderfrekvenser som ble skilt fra hverandre så godt som mulig med oppsettet ved testing. De to nye frekvensene som ble vurdert som de mest egnede senderfrekvensene var dermed 39.0 kHz og 40,3 kHz. Dette er da med filtre som er beregnet for senterfrekvenser på 39,8 kHz og 41,8 kHz.

Etter innjusteringen av forsterkerne ble forskjellen i mottatt signal og ikke mottatt signal som lest av i figur 5.48 på 2,5 V - 1,2 V = 1,3 V. Dette var da med 1cm avstand mellom ultralydsenderen og ultralydmottakeren.

Avstanden mellom ultralydsenderen og ultralydmottakeren ble deretter økt til 2 cm, og nye tester ble gjort for å se hvordan det ville fungere med



Figur 5.49: Kanalseparasjon med justert forsterkning. Avstand 2 cm.

en større avstand. Også her ble forsterkningen til motstandene justert for å få best mulig kanalseparasjon og forskjell i mottatt og ikke mottatt signal. Etter uttesting ble det beste resultatet med 2 cm mellom sender og mottaker som vist i figur 5.49. Ved måling av de to potensiometrenes motstand ble den målt til å være 749,5 kΩ for 39kHz-inngangen, og 825,9 kΩ for 40,3 kHz-inngangen. 40,3 kHz-inngangen har et noe lavere utgangspunkt enn 39,0 kHz-inngangen er det naturlig at forsterkningen må ligge høyere. Når avstanden mellom ultralydsenderne var på 1 cm, kunne forsterkningen legges på samme nivå, da forsterkerne gikk i metning ved frekvensene.

Ved testing med 5 cm avstand mellom sender og mottaker, var det ikke mulig å skille de to kanalene fra hverandre på en god måte.

Ved 5 cm avstand mellom senderen og mottaker kan altså ikke de to signalene skilles ifra hverandre lengre, noe som er mange ganger kortere avstand enn antatt. Når figur 5.22, som viser transdusernes ulastede frekvensrespons, sammenlignes med figur 5.23, som viser transdusernes

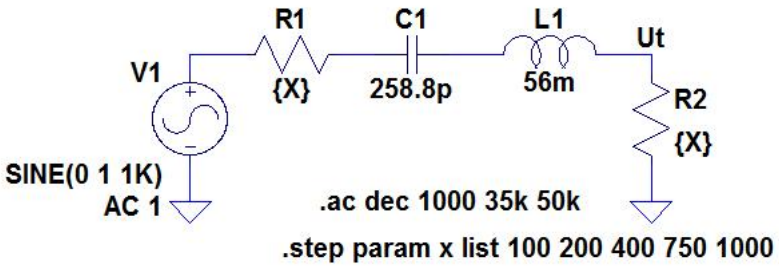
	Forsterkning 1 cm	Forsterkning 2 cm
39.8kHz	16,7	25,0
41.8kHz	16,7	27,5

Tabell 5.3: Forsterkning i forsterkerne.

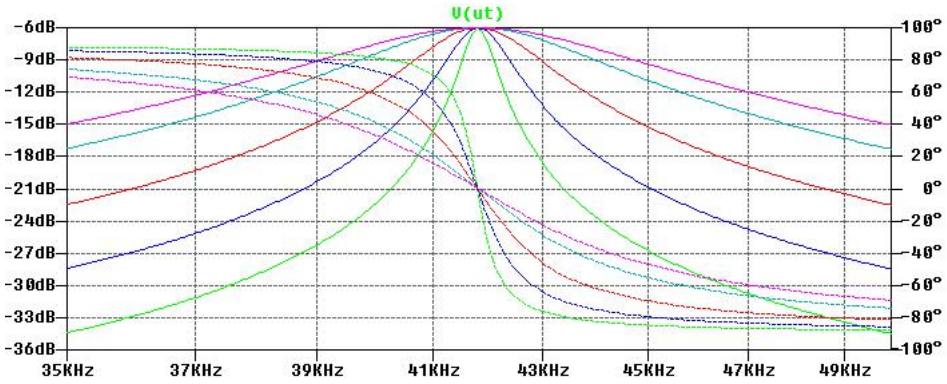
lastede frekvensrespons, vises det at når mottakeren blir belastet med de to parallelle filtrene vil signalet dempes med så mye som 25dB i passbåndet. Filtrene trekker altså mer energi fra ultralydmottakeren enn den kan levere. Dermed blir signalet betydelig dempet, og maksimal avstand mellom sender og mottaker betraktelig forkortet. For å løse dette problemet kan det gjøres forskjellige tiltak, men det er ikke tvil om at ultralydmottakeren ikke kan drive lasten filtrene representerer.

En mulig løsning på dette problemet kan være å øke inngangs- og utgangsmotstanden til filteret, da økt motstand vil senke lasten på mottakertransduseren. Desverre er det med formel 5.27 vist at med økt inngangs- og utgangsmotstand på filteret vil filteret bli mindre skarpt, og eventuelt kreve bruk av enda større spole for å opprettholde filterets skarphet. Enda større spoler enn det som allerede er i bruk vil gi enda større usikkerhet og ulinearitet, og det er en fordel om dette kan unngås. Det ble derfor gjort en simulering på filterets skarphet der inngangs- og utgangsmotstanden gradvis ble økt. Som vist i figur 5.50, med resultat i figur 5.51, ble det simulert med inngangs- og utgangsmotstand på 100Ω (grønn kurve), 200Ω (blå kurve), 400Ω (rød kurve), 750Ω (turkis kurve) og 1kΩ (lilla kurve). Som antatt blir filtrene mindre skarpe med økt inngangs- og utgangsmotstand. Likevel, med økt motstand vil selve mottakertransduseren belastes mindre, og dermed vil den kunne opprettholde frekvenskarakteristikken sin bedre, noe som kommer kanalseparasjonen til gode. Ved å øke utgangsmotstanden vil de to motstandene som gir signaljord også bli tilsvarende større. Dette vil som en positiv bieffekt senke det totale strømtrekket i kretsen.

En annen løsning kan være å sette inn en operasjonsforsterker som en spenningsfølger [25] eller eventuelt med lav forsterkning på ultralydmottakerens utgang, før filtrene. Da en operasjonsforsterker ideelt har en uendelig inngangsresistans og greit kan levere opp mot 20 mA på utgangen [24, 28] Kan en operasjonsforsterker fungere som et buffer. Signalet fra ultralydmottakeren vil dermed ikke dempes som følge av at den belastes for mye. Signalet vil deretter drives av operasjonsforsterkeren, når denne er koblet som en spenningsfølger med negativ tilbakekobling. Operasjonsforsterkeren kan drive en mye større last enn mottakertransduseren klar-



Figur 5.50: Filtersimulering med variabel inngangs- og utgangsmotstand.



Figur 5.51: Resultat av filtersimulering med variabel inngangs- og utgangsmotstand.

er. Denne løsningen krever en brikke med to operasjonsforsterkere, da alle eksisterende operasjonsforsterkere allerede er i bruk. Dette vil medføre det ekstra strømtrekket en operasjonsforsterker medfører, og kan være vel anvendt bruk av strøm. For å skille de to inngangsfiltrene ifra hverandre kan det eventuelt brukes en operasjonsforsterker på hvert av de to filtrene. Dette vil gi et høyohmig skille mellom de to filterinngangene, og resonnans i den ene inngangen vil ikke påvirke den andre inngangen i nevneverdig grad.

Nok et alternativ som kan tenkes er å bytte ut det serielle LC-filteret med et parallell LC-filter som har en høyere inngangs- og utgangsmotstand. Da et LC-filter drar nytte av en høyohmig inngang og utgang vil lasten på mottakertransduseren, takket være disse høyohmige motstandene, bli mindre. Desverre ble det ved test vist at de parallelle LC-filtrene demper inngangssignalet med rundt 20 dB i forhold til et serielt LC-filter. Dermed blir resultatet med et parallelt LC-filter sammenlignbart med resultatet for det serielle LC-filteret.

Da tiden ikke strakk til til å produsere nye kretskort har ikke de alternative forbedringsforslagene blitt utprøvd i praksis.

Kapittel 6

Diskusjon

Overgang fra radio til ultralyd som kommunikasjonsbærer vil ha begrensninger i et sensornettverk, først og fremst med tanke på direktivitet. Mens den innebygde radioantennen er tilnærmet rundtstrålende [12], er ultralydsenderens og -mottakerens utsendte signal, som vist i figur 5.20 og figur 5.21, betydelig dempet ved omlag 50° avvik fra direkte kommunikasjonsvei. Det betyr at ultralydsenderen og -mottakeren i et kommunikasjonsnett vil måtte plasseres i en tilnærmet fast retning i forhold til hverandre, både horisontalt og vertikalt for å sikre kontakt mellom de to. Hvis ultralydkommunikasjonen skal gjøres rundtstrålende, på tilnærmet samme måte som radioantennen, kan det monteres flere transdusere på sender- og mottakerkortene. En vinkel på 90° mellom hver transducer vil gi en tilnærmet rundtstrålende ultralydsender og -mottaker. En todimmensjonal konfigurasjon vil dermed måtte benytte fire transdusere, mens en tredimmensjonal konfigurasjon vil måtte benytte seks transdusere, for å fungere rundtstrålende.

På sendersiden bør alle ultralydsenderne kunne drives av den samme boostkonverteren og sendertrinnet, men som en følge vil det trekke seks ganger så mye effekt fra boostkonverterens lagringskondensator. Oppladningstiden vil derfor også blir seks ganger så lang, noe som vil redusere båndbredden. Dette kan til en viss grad motvirkes ved å bruke en større lagringskondensator i boostkonverteren, da denne vil ha større effektoverskudd. Da det er vist at boostkonverterens svitsjefunksjon er den delen av senderkortet som trekker absolutt mest strøm vil overgangen til 6 ultralydsendere i følge målingen med en ultralydsender øke strømtrekket med 6 mA * 5 = 30 mA. Når senderkortet kontinuerlig trekker en strøm på 125 mA, som målt, betyr det at Tmote Skys standard strømforsyning, to seriekoblete AA-batterier på typisk 1000mAh [39], vil brukes opp på $\frac{1000mAh}{125mA}$= 8 timer. I tillegg må strømtrekket til Tmote

Sky sensornoden, om lag 1,8mA (MCU aktiv, radio avslått), som styrer senderkortet legges til.

På mottakersiden kan hver ultralydmottaker monteres med egne filtre, eller de kan kobles til en felles inngang før filtreringen av signalet. Det er usikkert hvordan flere parallelle ultralydmottakere vil påvirke hverandre, dette må derfor tas hensyn til og undersøkes før de kobles til felles filtre. Mottakerkortet vil typisk bruke opp Tmote Skys standard strømforsyning på $\frac{1000mAh}{14mA}$= 71 timer, eller om lag tre dager. I tillegg må strømtrekket fra Tmote Sky sensornoden på 1,8 mA legges til.

En Tmote Sky som driver både et sender- og et mottakerkort har med andre ord en batterilevetid på $\frac{1000mA}{1{,}8mA+125mA+14mA}$ = 7 timer.

Kapittel 7

Konklusjon

Denne oppgaven har sett på hvordan overgang fra radiokommunikasjon til ultralydkommunikasjon kan implementeres i et sensornettverk, med begrenset strømforsyning, med tanke på bruk i sensornettverk under vann. Best egnede modulasjonsmetode er vurdert, og deretter er eksterne utbyggingskort til Tmote Sky noder for sensornettverk konstruert og utprøvd. Det er tydelig at en stor begrensning ved ultralydkommunikasjon er behovet for høye spenninger til ultralydsenderen. Det å generere 30 V spenning ut fra en 3 V spenningskilde viser seg å være en meget energikrevende oppgave som tømmer en standard 3 V strømforsyning bestående av 2 AA-batterier på få timer.

I mottakerdelen gir analoge filtre problemer med impedansetilpasning mellom ultralydmottaker og kretsen ellers, men forslaget til forbedring av mottakeren, med en operasjonsforsterker som buffer, bør kunne hjelpe på dette, og dermed øke kommunikasjonskanalens rekkevidde.

TinyOS' tidsoppløsning fører med seg mye ekstra elektronikk på sendersiden, som igjen begrenser hastigheten til systemet, og må derfor kunne sies å være en stor begrensning i kommunikasjonssystemet. Det er likevel vist at ultralyd kan fungere som kommunikasjonsbærer over korte avstander i et sensornettverk, hvis elektronikken rundt legges til rette for dette. Ultralyd kan derfor ha potensiale til å være en fullgod kommunikasjonsbærer i et sensornettverk hvis strømtrekket kan reduseres og lavere belastning av ultralydmottakeren øker avstanden på kommunikasjonskanalen.

7.0.11 Videre arbeid

Etter at designet var utprøvd, ble det klart at designet har et forbedringspotensiale i form av impedansetilpasning mellom filtrene

og ultramottakeren, og i å få ned strømforbruket. Utprøving med god impedansetilpasning mellom analogt filter og ultralydmottaker bør derfor gjøres. Generering av signaljord står også for en stor del av det totale strømtrekket på mottakersiden. Dette kan med fordel gjøres på en mer effektiv måte.

Da TinyOS operativsystemet har såpass mye større begrensninger enn først antatt kan introduksjon av en mikrokontroller og overgang til digital signalprosessering hos både sender og mottaker være verdt å se nærmere på. En mikrokontroller kan da eventuelt også avlaste Tmote Skys innebygde mikroprosessor og samtidig gi bedre tidsoppløsning i systemet. En mikrokontroller kan muligens også dra nytte av mer effektive modulasjonsmetoder enn analog elektronikk, og bør derfor studeres videre.

Kapittel 8

Bibliografi

Bibliografi

[1] Jaeger, Richard C. Microelectronic circuit design. s.577-578. USA: McGraw-Hill 2008.

[2] Sverre Holm, Professor ved UiO.

[3] Horowitz, Paul. The Art of electronics 2nd edt. s.267. USA: Cambridge University Press. 1989.

[4] Maxim. Aksess: http://www.maxim-ic.com/appnotes.cfm/appnote_number/737

[5] Ingebrigtsen, Rolf. Analoge kretser og komponenter. Innføring i elektronikk for ingeniører. s.3.20. Norge: Høyskoleforlaget. 2001.

[6] Ingebrigtsen, Rolf. Analoge kretser og komponenter. Innføring i elektronikk for ingeniører. s.3.21. Norge: Høyskoleforlaget. 2001.

[7] Stallings, William. Data and computer communications 7th edt. s.142-152. USA: Pearson Prentice Hall. 2004

[8] Williams, Arthur B. Taylor, Fred J. Electronic Filter design Handbook, 3rd edition. s.5.7-5.27. USA: McGraw-Hill inc. 1995.

[9] Phiilips NXP 74hc132 schmitt trigger. Datasheet
Aksess: http://www.farnell.com/datasheets/43297.pdf

[10] Floyd, Thomas L. Principles of Electronic Circuits, Conventional Current Version, int. edt. 7th edt, s.773-781. USA: Pearson Prentice Hall 2003.

[11] Moteiv Community, Connecting external sensors.
Aksess: http://www.moteiv.com/community/Connecting_External_Sensors

[12] Moteiv Corporation, Tmote Sky Datasheet. Aksess: http://www.sentilla.com/moteiv-endoflife.html

[13] The nesC Language: A Holistic Appoach to Networked Embedded Systems.
Aksess: http://nescc.sourceforge.net

[14] Levis, Phillip. TinyOS Programming. June 28, 2006.
Aksess: http://csl.stanford.edu/~pal/pubs/tinyos-programming.pdf

[15] TinyOS Extension Proposal 114 (TEP114).
Aksess: http://tinyos.cvs.sourceforge.net/*checkout*/tinyos/tinyos-2.x/doc/html/tep114.html

[16] Wikipedia. Aksess: http://en.wikipedia.org/wiki/555_timer_IC

[17] FAIRCHILD SEMICONDUCTOR HUF75307D3 NMOS transistor. Datasheet
Aksess: http://www.fairchildsemi.com/ds/HU%2FHUF75307D3.pdf

[18] INTERNATIONAL RECTIFIER IRFU9024NPBF, PMOS transistor. Datasheet
Aksess: http://www.irf.com/product-info/datasheets/data/irfr9024n.pdf

[19] ON SEMICONDUCTOR BC373G NPN-transistor. Datasheet
Aksess: http://www.farnell.com/datasheets/82831.pdf

[20] ON SEMICONDUCTOR BC556BZL1G PNP-transistor. Datasheet
Aksess: http://www.onsemi.com/pub_link/Collateral/BC556B-D.PDF

[21] Floyd, Thomas L. Principles of Electronic Circuits, Conventional Current Version, int. edt. 7th edt. s. 525. USA: Pearson Prentice Hall 2003.

[22] LTspice/SwCAD III:
Aksess: http://www.linear.com/designtools/software/switchercad.jsp

[23] ST Microelectronics TS912. Datasheet
Aksess: http://www.farnell.com/datasheets/63486.pdf

[24] Horowitz, Paul. The Art of electronics 2nd edt. s.177. USA: Cambridge University Press. 1989.

[25] Horowitz, Paul. The Art of electronics 2nd edt. s.179. USA: Cambridge University Press. 1989.

[26] Hanssen, Leif. Wireless Sensor Network - Trådløse sensornettverk. FFI/Rapport-2006/04014.

[27] Horowitz, Paul. The Art of electronics 2nd edt. s286-291. USA: Cambridge University Press. 1989.

[28] Ingebrigtsen, Rolf. Analoge kretser og komponenter. Innføring i elektronikk for ingeniører. s. 9.15 Norge: Høyskoleforlaget. 2001.

[29] Horowitz, Paul. The Art of electronics 2nd edt. s.229-231. USA: Cambridge University Press. 1989.

[30] TinyOS Help, interface TMicro.
Aksess: http://www.mail-archive.com/tinyos-help@millennium.berkeley.edu/msg12078.html

[31] Murata MA40S4R spesifikasjoner.
Aksess: http://search.murata.co.jp/Ceramy/CatalogAction.do?sHinnm=? &sNHinnm=MA40S4R&sNhin_key=MA40S4R&sLang=en&sParam=ma40s4

[32] Murata MA40S4S spesifikasjoner.
Aksess: http://search.murata.co.jp/Ceramy/CatalogAction.do?sHinnm=? &sNHinnm=MA40S4S&sNhin_key=MA40S4S&sLang=en&sParam=ma40s4

[33] IEEE 802.15.4 kommunikasjonsstandarden.
Aksess: http://www.ieee802.org/15/pub/TG4.html

[34] National Instruments Developer Zone. SPICE Simulation overview.
Aksess: http://zone.ni.com/devzone/cda/tut/p/id/5414

[35] TinyOS.net TEP2, Hardware Abstraction Architecture.
Aksess: http://www.tinyos.net/tinyos-2.x/doc/html/tep2.html

[36] Preisig, James. Acoustic propagation considerations for underwater acoustic communications network development. ACM SIGMOBILE Mobile Computing and Communications Review. vol. 11, Issue 4. s.2-10.

[37] Lurton Xavier, An Introduction to underwater Acoustics. s.19-21. Berlin: Springer-Verlag. 2002.

[38] Mohan, Ned, m.fl. Power Electronics, converters, applications, and design. 3rd edition. s.172-178 USA: Wiley. 2003

[39] Wikipedia, AA battery. Aksess: http://en.wikipedia.org/wiki/Aa_battery

[40] Wikipedia, Diode bridge. Aksess: http://en.wikipedia.org/wiki/Diode_bridge

Kapittel 9

Appendix

9.1 Komponentliste

9.1.1 Mottakerkort

- 1 stk ultralydmottaker, Murata MA40S4R
- 2 stk Motstand 100 Ω
- 2 stk Spole 56 mH
- 4 stk Motstand 200 Ω
- 2 stk Kondensator 220 pF
- 1 stk Kondensator 39 pF
- 1 stk Kondensator 66 pF
- 4 stk Motstand 3 kΩ
- 2 stk Potensiometer 1 MΩ
- 2 stk Motstand 30 kΩ
- 2 stk Operasjonsforsterker
- 2 stk Diode
- 2 stk Kondensator 10 nF
- 4 stk Potensiometer 500 kΩ
- 1 stk standard 10-pins konnektor

9.1.2 Senderkort

- 1 stk ultralydsender, Murata MA40S4S
- 1 stk Kondensator 10 μ F
- 3 stk NPN transistor
- 2 stk PNP transistor
- 2 stk Diode
- 5 stk Kondensator 10 nF
- 1 stk Spole 4,7 mH
- 4 stk Potensiometer 10 kΩ
- 1 stk Potensiometer 500 kΩ
- 1 stk 74HC00 NAND-gate
- 1 stk 74HC03 Inverter.
- 1 stk 10-pins konnektor.

9.2 programkode

9.2.1 Senderkort

Konfigurasjonsfil

```
configuration SendAppC

{
}

implementation
{
 components MainC, SendC, LedsC, HplMsp430GeneralIOC;
 components new TimerMilliC() as Timer0;
 components new TimerMilliC() as Timer1;
 components new TimerMilliC() as Timer2;
```

```
 SendC -> MainC.Boot;

 SendC.Port1 -> HplMsp430GeneralIOC.Port35;
 SendC.Port0 -> HplMsp430GeneralIOC.Port34;
 SendC.Boost -> HplMsp430GeneralIOC.ADC7;

 SendC.Timer0 -> Timer0;
 SendC.Timer1 -> Timer1;
 SendC.Timer2 -> Timer2;
 SendC.Leds -> LedsC;
}
```

Modulfil

```
#include "Timer.h"



module SendC
{
  uses interface Timer<TMilli> as Timer0;

 uses interface Timer<TMilli> as Timer1;
  uses interface Timer<TMilli> as Timer2;

  uses interface HplMsp430GeneralIO as USPort0;

  uses interface HplMsp430GeneralIO as USPort1;

  uses interface HplMsp430GeneralIO as USBoost;



  uses interface Leds;
  uses interface Boot;
}
implementation
{
int i = 0;
int counter;
int data;
```

```
 event void Boot.booted()
  {
  call USBoost.makeOutput();
  call USBoost.set();
  call USPort0.makeOutput();
    call USPort0.clr();
call USPort1.MakeOutput();
call USPort1.clr();
counter = 0;
data= 0;
  call Timer0.startPeriodic(1000);
  }

  event void Timer0.fired()
  {
  counter++;
  data = counter;
  call Leds.led.Toggle();
  call Timer1.startPeriodic(10)
  }

  event void Timer1.fired()
  {
  i++;
  if (data & 128)
  {
  call USPort0.set();
  call Timer2.startOneShot(1);
  }
  else
  {
  call USPort1.ser();
  call Timer2.startOneShot(1);
  }
  data = data << 1;

  if (i>7)
  {
  call Timer1.stop();
  i = 0;
```

```
  }
  }

  event void Timer2.fired();
  {
  call USPort0.clr();
  call USPort1.clr();
  call Timer2.stop();
  }
}
```

9.2.2 Mottakerkort

Konfigurasjonsfil

```
configuration MottaAppC
{
}
implementation
{
components MainC, MottaC, LedsC, HplMsp430GeneralIOC, HplMsp430InterruptC;
components new TimerMilliC as Timer0;
components new TimerMilliC as Timer1;

MottaC -> MainC.Boot;

MottaC.Lav -> HplMsp430GeneralIOC.Port20;
MottaC.LavInput -> HplMsp430InterruptC.Port20;
MottaC.Hoy -> HplMsp430GeneralIOC.Port21;
MottaC.HoyInput -> HplMsp430InterruptC.Port21;

MottaC.Timer0 -> Timer0;
MottaC.Timer1 -> Timer1;
MottaC.Leds -> LedsC;
}
```

Modulfil

```
#include "Timer.h"

module MottaC
```

```
{
uses interface Timer<TMilli> as Timer0;
uses interface Timer<TMilli> as Timer1;

uses interface Leds;
uses interface Boot;

uses interface HplMsp430GeneralIO as Lav;
uses interface HplMsp430Interrupt as LavInput;
uses interface HplMsp430GeneralIO as Hoy;
uses interface HplMsp430Interrupt as HoyInput;
}
implementation
{
int one;
int i;
int j;
int counter;

event void Boot.booted()
{
TOSH_MAKE_ADC0_INPUT();
TOSH_MAKE_ADC1_INPUT();

one = 0;
i = FALSE;
j = FALSE;
counter = 0;
call HoyInput.enable();
call LavInput.enable()
call Timer1.startPeriodic(1000);
}

event void Timer0.fired()
{
i = FALSE;
j = FALSE;
}

event void Timer1.fired()
{
```

```
if(counter & 0x1)
{
call Leds.led0On();
}
else
{
call Leds.led0Off();
}
if(counter & 0x2)
{
call Leds.led1On();
}
else
{
call Leds.led1Off();
}
if(counter & 0x4)
{
call Leds.led2On();
}
else
{
call Leds.led2Off();
}
}

async event void HoyInput.fired()
{
atomic
{
i = TRUE;
if (i=TRUE)
{
call Timer0.startOneShot(4);
}
else
{
counter = counter << 1;
counter = counter|1;
}
}
```

```
}
async event void LavInput.fired()
{
atomic
{
j = TRUE;
if (j=TRUE)
{
call Timer0.startOneShot(4);
}
else
{
counter = counter << 1;
counter = counter|0;
}
}
}
}
```